Semantification of Text through Summarisation

By

Monika Joshi (M. Tech., Computer Applications)

A THESIS PRESENTED IN FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF DOCTOR OF PHILOSOPHY

School of Computing and Engineering

Ulster University, Shore road, Jordanstown

February 2019

I confirm that the word count of this thesis is less than 100,000 words

Contents

List of Tables

List of Figures

Acknowledgements

I would like to thank my supervisors Prof. Hui Wang and Prof. Sally McClean for guiding me throughout my PhD. The encouragement, motivation and supervision provided by them during my stay at university and while working from my home country are invaluable. They are the pillar of strength and support for me.

I would also like to express my gratitude towards my brother, my mother and my husband for their unconditional love. I am grateful to my friends who are equal to family members and have always made their presence felt during this long journey.

Special thanks to my father Shri Umakant Joshi, who made me believe in my capabilities and encouraged me to dream big. He is the wind beneath my wings.

Abstract

The research topic of this thesis is semantic representation of text document and abstractive summarisation. Designing a semantic representation of text document is an important research topic due to increasing unstructured textual information over web. To automatically process this textual information first it should be represented in a standard way. In addition, abundance of information has increased demand for shortening lengthy online text documents from different genre i.e. patent documents, news articles into useful summaries.

In this thesis, we present a systematic analysis of different semantic representations of text data. We have analysed two ways of constructing semantic graphs from the semantic relations of words. One graph is based on logical triples of *subject-predicate-object* and the other graph is based on dependencies other than logical triples. Our experiments on benchmark datasets for text summarisation confirmed the effectiveness of new proposed graph in text summarisation.

We have also looked beyond traditional representations and proposed inclusion of object-oriented principles into semantic graph design. This resulted in *object oriented semantic graph* of text document where important entities of text are projected as object and different properties of objects are extracted from text by utilising different *natural language processing (*NLP) processes. Further methodologies were developed to generate abstractive summary directly from this graph instead of the original document. We have analysed the abstractive summaries generated from object-oriented semantic graph by automated evaluation tool ROUGE and by manual evaluation. Although the ROUGE results achieved by object-oriented semantic graph could not surpass the states of the art that were achieved by

extractive summarisers but results were better than previous semantic graph based summarisation results.

An analysis was done on inclusion of various syntactic units into summary and the conclusion of this analysis is that including adjectives into summary improves the informativeness of summary, but inclusion of adverbs does not affect it. Overall, this thesis presents a theory and methodology to generate efficient semantic graphs from text document and gives strategies to use this graph as a replacement for original document in NLP processes such as text summarisation.

The research work presented in this thesis can be extended further by improving the graph generation capabilities to handle texts that are more complex and by improving the ranking methodologies for different graph elements. Quality of abstractive summaries generated from object-oriented semantic graph can be improved by including better natural language generation techniques.

Abbreviations

AMR	Abstract meaning representation
AR	Anaphora resolution
DUC	Document Understanding Conference
HITS	Hyperlink-Induced Topic Search
HMM	Hidden Markov Model
ILP	Integer Linear Programming
IR	Information radius
JUNG	Java Universal Network/Graph Framework
LDA	Latent Dirichlet allocation
LSA	Latent Semantic Analysis
NER	Named Entity Recognition
NLP	Natural Language Processing
NMF	Non-negative matrix factorization
NP	Noun Phrase
OTS	Open Text Summariser
O-O Semantic Graph	Object Oriented Semantic Graph
PLSA	Probabilistic Latent Semantic Analysis
POS	Part of Speech
PP	Prepositional Phrase

QA	Question Answering
RDF	Resource description framework
ROUGE	Recall-Oriented Understudy of Gisting Evaluation
SF	Semantic frame
Stanford CoreNLP	Suite of core NLP tools from Stanford
SVD	Singular value decomposition
SVM	Support vector machine
SVO	Subject-Verb-Object triple
TAC	Text analysis conference
UD	Universal dependency
UML	Unified modelling language
UNL	Universal network language
VP	Verb Phrase
WSD	Word sense disambiguation

Chapter 1

Introduction

1.1 Background

Information technology has increased the mediums to share information and this has in turn increased the access to quality useful information tremendously. Online textual information is one of such information resource which is abundantly present in the form of news articles, blogs, social media interaction, twitter posts, online articles in journals, conference proceedings etc. Users are accessing this data in academia, business or daily tasks to make informed decisions for various purposes. The amount of data in these resources is large and it makes it difficult to manually analyse the available textual data for different usages. Online searches bring out list of documents, but going through each document thoroughly before deciding the most relevant document is a tedious task. Similarly, to understand user perception about different products and its features, going through all reviews is not possible. This necessitates the automated processing of these text documents to provide a concise report or review which should indicate the gist of information present there. Among several of such automated text processing tasks the few are known as single document summarisation, multiple document summarisation, opinion summarisation, question answering, and sentiment analysis. We have focussed on the specific task of generating a concise informative summary from a big single document, *automatic single document summarisation*.

This task has been extensively studied in natural language processing and computational linguistics. This is used in automatic abstract generation for patent papers, research papers, news highlights generation and used in assisting query search tasks. Text summarisation can reduce the processing time of other NLP systems, by producing a shorter substitute for original text. Its known applications are in *information retrieval*, *text classification* and *question answering*. In information retrieval, documents are indexed by its relevance to the queried event/topic from many data sources. Using summary instead of the whole document improves the indexing time for the queried documents in general information retrieval and Geo specific information retrieval [1]. In medical domain, the information retrieval varies according to the user requirements such as physician access clinical data records to analyse and correlate diseases whereas patients require access to current treatment plans and medicines. In addition, information presented to patient needs to be less domain specific. Text summarisation has been utilised in presenting clinical information to cater to different user needs [2, 3, 4].

Question answering (QA) task is another NLP process to be profited from summarisation task. In cases where many documents are found to be giving relevant answers to a non-factoid question, a fusion summary of all information will produce a more informative answer. Using query focussed summaries for web semantic QA has improved performance in finding correct answers over direct QA on the internet snippets [5]. Location specific summaries can collect information about the climatic conditions and geographical conditions of a specific region. These summaries have been utilized for creating an information portal of different regions and for climate change analysis [6].

Another interesting application of Summarisation is generating a headline or smaller version of news or emails to be displayed on the smaller screen of personal digital assistants while keeping the information loss minimum [7]. Text summarisation has shown to be effective in

producing catchy headlines for Automatic tweet generation[8] in news genre. Thus, text summarisation has a good scope to be combined with other NLP applications. In some cases, it can improve the time performance of these NLP applications by reducing the content to analyse from a complete document to a summary.

1.2 Motivation

In our research, we have aimed to develop new methodologies to the automatic text summarisation research. Our work is focussed on understanding text document in an automatic way, representing the understanding in a well-defined structure and finally generating summary from this intermediate structure. This approach follows the well accepted definition of automatic summarisation given by K. S. Jones [9]. According to the definition, summarisation is a three-step process interpretation, transformation and generation. In interpretation step the machine should be able to understand the document and represent it, resulting in a source text representation often described as semantic representation. Later transformation and generation of summary are done on this semantic representation. Semantics in linguistic relates to study of meaning in linguistic expressions. Semantic representation describes the structure of semantic and contextual associations of meaning of words to other words and words to concepts in the linguistic documents[10]. A semantic representation should unambiguously represent the information present in the document. The coverage of information and the ease to access and infer further information determines the efficiency of a semantic representation. There are automatic summarisation approaches where the first step is not given explicit consideration such as extractive summarisation where the summary is generated by identifying summary sentences in the original text document. This kind of summarisation is most researched and has shown good

performance in evaluation tasks organised by *National Institute of Standard and Technology* (NIST). There are also extractive-abstractive summarisers that extract sentences from original document and before or afterwards apply sentence reduction techniques to shorten the summary sentences. Literature indicates that research in extractive summarisation has been exhausted and has reached a performance limit which cannot be improved much beyond the current maximum ones[11–13]. Also extractive summaries suffers the lack of coherence due to unresolved co-references and wrong links between extracted sentences[14]. In contrast, the abstractive summarisation, which follows the complete three-step process to generate summary, is relatively very less researched. The performances of some of the similar kind abstractive summarisers are not tested on standard datasets, so comparative evaluation is difficult. Other reported results based on these summarisers are comparative but not better than extractive methods. Nevertheless, abstractive summarisation is quintessential in different genre of summarisation. One of that is opinion summarisation because selecting sentences as done in extractive summarisation may give misleading information if the sentences present controversial opinions[15]. In this case, an abstractive summary that has been constructed from the interpretation of all opinions will bring out the gist of overall opinions. In single document summarisation domains where the information is less controversial such as in news article domain[16], the gain of abstractive summariser over extractive is more diverse information by including shortened sentences and better coherence in the summary by removing the dangling references issues in extractive summaries[17]. Thus, we have broken down our goal of automatic text summarisation into sub problems of semantic representation and generation of abstractive summary from semantic representation. In next Section, we describe our research project plan to achieve the set objective.

1.3 Aim and objective of research

This project studies abstractive text summarisation and provides the foundation for the development of effective abstractive text summarisation systems.

The objectives of our research are:

1. Design semantic representation for the online text document.

2. Develop a tool to generate the semantic representation

3. Design methodology to construct summary from semantic representation.

This Chapter provides an overview and context of the PhD project. A detailed literature review of text summarisation is given in Chapter 2 with a particular focus on abstractive text summarisation. Also in literature review, we have shown existing semantic representations of text document and their usage in automated text summarisation.

Chapter 3 describes some exploratory studies in this research project. We started with the sentence alignment task and contributed a better performing method that is based on an extension of earlier methods. Sentence alignment is a natural language processing task to identify similar part of two or more sentences, which is most used in machine translation, but also used in summarisation for redundancy removal. To reduce the time complexity of existing tree based bottom-up alignment method we have proposed to divide sentences into clauses and then form sentence alignment from clause alignment.

Chapter 4 describes the next study to compare two types of summarisation based on semantic graphs *triple based semantic graph* and *dense semantic graph*. In this study we have proposed a new semantic graph to improve the performance of summarisation compared to triple based summarisation.

In Chapter 5 we have described the enhancement of dense semantic graph to *object-oriented semantic graph*. This Chapter contains the rules to generate the graph from a text document with corresponding examples and describes the cases that cannot be handled by this design.

In Chapter 6 we have shown the methodology to construct abstractive summary from object-oriented semantic graph. We have compared the summaries generated from object-oriented semantic graph with previous dense graph based summaries and also analysed results of this summariser in other domains such as medical text domain..

Chapter 7 present the conclusion from all studies and gives direction for future work.

1.4 Published work

We have disseminated the analysis and results from these studies in the following four research papers.

1. Effect of Clause Splitting on Sentence Alignment (Presented in IEEE 8th NLP-KE conference, published in *International Journal of Advanced Intelligence, vol. 5, 2013*)
2. Dense semantic graph and summarisation (Presented in DART-2013, AI-IA, published in book : *Emerging Ideas on Information Filtering and Retrieval, Cham, Switzerland:Springer International Publishing, pp. 55-67, 2018)*
3. Object-oriented semantic graph (Presented at MIKE-2014 conference, Published in book : *Mining Intelligence and Knowledge Exploration. Lecture Notes in Computer Science, vol 8891. Springer*)
4. Abstractive Summarisation for Single Documents – an Object Oriented Semantic Graph based Approach (Submitted to Journal- Computer Speech & Language, Elsevier B.V.).

1.5 Conclusion

Automatically generating a summary that is close to human written summary is an ambitious goal and any advancement in the direction of abstractive summarisation is step towards that goal. Through following a systematic research method we have worked to develop a summariser that can understand the semantics of text and then generate an optimum summary. The two main contributions of this research are: a) object-oriented semantic graph and b) the method to generate summary from a semantic representation can also be used in other NLP tasks and can be improved further by incorporating more rules obtained by analysing more text inputs.

Chapter 2

Text Summarisation – A Critical Review

Automatic text summarisation has been a research area since 1950's. It has gained popularity again after the information technology revolution in post internet era. It is part of the wider research area of natural language processing and deep machine learning. Summary generation from a document requires understanding its structure and meaning. It also requires natural language generation capabilities to generate new readable text from information units. This complete research problem of text summarisation contains many sub problems such as *semantic representation of text*, *text generation* that are being researched independently across various NLP groups. In this Chapter we review various popular summarisation methods with more focus on those that are based on extracting deeper semantic relations hidden inside text, with the goal of placing our work in context and identifying research gaps. We review the different kinds of summaries produced by available summarisers and their uses in other NLP tasks. We also review the summary evaluation metrics, especially those that are used in our research work. We begin the review with a short overview of contemporary NLP research problems few of which have been utilised as subtasks in the process of automated text summarisation.

2.1 NLP research problems

NLP is a branch of artificial intelligence and computational linguistics research field with focus on developing techniques for natural language understanding, natural language generation and their application in human-computer interaction. NLP comprises of various

tasks such as tokenisation, syntax parsing, discourse parsing, co-reference resolution, named entity recognition, question answering, sentiment analysis, automatic text summarisation, machine translation etc. NLP task *automatic text summarisation*, which is the topic of our research, is focussed on shortening a text document or multiple similar documents into a smaller consolidated version. Automatic text summarisation utilises techniques developed in other related NLP tasks to understand and analyse the text document thus first we briefly look back at other related NLP research problems. Lexical analysis of text includes tasks for recognising sentence boundaries, tokenising sentences into words and syntactic parsing of text includes task for generating grammatical structure of sentences in tree form using the context free grammar rules [18]. First implementation of lexicalizer and parser was lex/Yacc. From the integration of static and machine learning approaches major improvement was achieved in the accuracy of probabilistic parsers and that led to their widespread usage in other NLP techniques[19].

In word sense disambiguation (WSD) task techniques are developed to identify the intended meaning of word/phrase from the context of its usage. State of the art WSD systems are supervised systems [20] and knowledge base systems [21]. Inclusion of WSD has shown improvement in summarisation[22] and semantic graph generation approaches[23].

To recognise the relation among roles of different words of a sentence with respect to predicates the dependency parsing task is utilised. Stanford's phrasal structure based dependency parser is the best performing dependency parser and most utilised in academics and Industry based research [24]. *MaltParser* is a discriminative dependency parser trained on treebank and has better speed with slight trade-off of accuracy compared to other parsers[25].

The task utilised to label the words in text that are name of a person, place or organisation is called Named Entity Recognition (NER) [26]. Best performing NER system on the shared

task of *Conference on Computational Natural Language Learning-2003* (CoNLL-2003) for English corpus is Illinois Named Entity Tagger[27,28]. It uses non-local features of text and knowledge derived from Wikipedia knowledge base to predict the NER labels of words. Stanford's NER system is also among the top performing systems and its *conditional random field* based models can be trained on any language corpus thus this NER system is available for many other languages such as German and Chinese[29].

Another task that is useful in automatic text summarisation and other NLP processes is co-reference resolution. Focus of this task is to resolve the references made by pronoun and noun words to recognised named entities. Best performing system by Lee et al. [30] in CoNLL-2011 shared task was based on rules and linguistic information and it is one of the best publicly available co-reference resolver. The system by Fernandes et al.[31] , which utilised structured perceptron algorithm on latent constrained structures gave the best performance on CoNLL-2012 closed track. Chang et al. [32] has reported improved accuracy on CoNLL-2012 shared task corpus. Jointly solving co-reference resolution and NER tasks has shown good performance by reducing the errors in both tasks[33].

Sentence alignment in monolingual corpora is another important NLP task used for finding the related parts of two sentences. Best aligner approaches include word alignment approach by Yao et al. [34] which solves the alignment as sequence labelling problem. Among other best aligners are phrase based aligner by MacCartney et al. [35] and its faster version developed by Thadani et al. [36] by solving it as *Integer Linear programming* (ILP) problem using dependency relations and other contextual features.

Textual entailment/paraphrasing tasks are researched for deciding whether the information present in small text snippet can be inferred from bigger text/paragraph[37]. Textual entailment techniques has wider used in automatic text summarisation[38]. Machine learning

based approaches which utilises lexical features and dependency relations have achieved best accuracies on the shared dataset of *Recognising Textual entailment shared task*[39,40].

Thus Semantic representation and text summarisation research incorporates techniques from other NLP research topics to improve information content of summary and reduce redundancy. Research in Summarisation has been gaining more attention due to the portable small screen devices i.e. smartphones, tablets becoming a common medium to surf information and due to the limits they set for display space of the information content. Also in other NLP processes such as *information retrieval* and QA, using document summaries instead of original document saves processing time and that in turn has increased focus on finding smaller substitute for bigger document. Overall Summarisation leads to various kinds of summaries as per the methods used and the user requirements that are discussed in detail in next Section.

2.2 Different types of summarisation

Auto text summarisation began with work on news articles and academic papers. Diversity of available textual resources has modified the initial aim of summarisation research to broaden its categories of summarisers according to text type and user requirements. Now the text domain is not limited to only news data and academic papers but it has been widespread to include all social media interaction, financial news, and patent documents. Here we will discuss the different kind of summaries that an automatic text summariser can generate.

2.2.1 Single-document/multi-document summary

Summaries can be generated from one document or many similar documents. First type is researched under single document summarisation field and latter one under multiple-document summarisation. The text summarisation methods differ for both types of summaries. In a single document, generally every sentence is unique and not repeated due to limited space whereas in many similar documents on same topic, information is repetitive. Hence the methods in single-document summarisation generally focus on ranking sentences for extracting it or on ranking smaller information units (i.e. phrases) for generating abstract e.g. abstract of a research paper. Whereas multiple document summarisation methods focus on combining information from many documents but at the same time preventing redundant information to be included in the summary. This makes position and discourse based approaches more popular in single document summarisation [41,42] whereas to combine information clustering based solutions are more appropriate for multi-document summarisation[43,44]. Although interchange of methodologies are common such as Fusion based methods are equally explored in both summarisations but has shown good results for multi-document summarisation[45].

2.2.2 Extract/abstract

The format of summary differs in the way it is constructed from original text. Most researched summaries are of extractive nature, where the final summary is made up of the few sentences taken from the documents without any modification to original text. Thus extractive summarisation concentrates on ranking sentences but not on natural language generation. Abstractive type of summaries has a modified text to connect the sentences in a better coherent way and many further improvement such as co-reference resolution to

resemble human authored summaries [46]. Abstractive summarisation is the long sought aim of researchers and work in this direction is often described as semi-abstractive due to very slight replacement or deletion of words in original sentences by compression and sentence simplification methods.

2.2.3 Generic/query-focused summary

Generic summaries present all the important information from the original document. It can also be used as a replacement of original document for a limited analysis. Query-focused summary presents information that a user requires from the document[47,48]. In this type of summary user provides a query and the summarizer gathers the important facts from the document that are relevant to that query. Other important information that is not relevant to the query is not included in the summary.

2.2.4 Indicative/informative summary

Any text document can be thought of as textual information distributed around some major topics and their subtopics. Indicative summary points the topics of document by either generating a headline for that article or by showing main key phrases. The main focus in this kind of summarisation is topic identification[49].

2.2.5 Opinion summarisation/product review

Opinion summarisation, which also falls under the broader field of text summarisation are, dedicated techniques, which analyses opinions of people. One example of this kind of summarisation is to understand public perception about some event and their action course from twitter/social media feed around the hashtag of that event. It is popular among security

services to understand the people anger or sympathy about current happenings. Also comes under this category review summarisation which analyses online user reviews about some products/movie by evaluating different aspects such as quality of product, durability or performance and direction in movie and gives collective rating of positive/negative about that aspect [50]. An opinion summariser consists of opinion mining and text summarisation. Text analysis conference (TAC) 2008 had a separate opinion summarisation task for evaluation and has received good participation [51].

2.2.6 Update summary

Document Understanding Conference (DUC)-2007 introduced a new task update summarisation. In this task two sets of documents are given as input to summariser. First set contains the documents to summarise as in the multi-document summarisation. Second set contains the documents that the user has already gone through and expect summary to have updated information about the already read document. This summary is called update summary and its practical application lies in summarising the answers form forums where the user has already gone through some answers and want new relevant information about the query. [9] proposed concept of *filtering features* to be included for generating update summaries[52].

2.2.7 Survey summary

Survey summary is a comparatively long summary of all relevant facts of a topic that fills the predefined template slots from a big corpus relevant to that topic i.e. literature review generation about a technical topic. The corpus for literature review can be automatically built by crawling all citations present in few seed article about the topic. Sauper&Barzilay [53]

described the application of survey summarisers to create a Wikipedia article about a technical topic, by manually developing some domain specific templates to be filled in by survey summariser. Another type of survey summary gaining attention is timeline summarisation where from past news data for a defined time period all the important events happening around a particular topic are summarised in chronicle order to give a detailed summary of facts[54]. This involves event detection, coherence structure generation and clustering facts to form a detailed summary[55].

All of the above mentioned summary types indicate that a summarisation system depends on various factors such as the input summary type, input language, user requirements, output length, output language etc. Considering these factors various summarisation approaches have been developed. A description of those approaches is given in Section 2.4. The next Section discusses the different representations of text that are used in different text summarizers as data structures.

2.3 Representations of linguistic information

To analyse the textual information at complete document level different representations has been constructed. Here we discuss the few utilised for text summarisation.

2.3.1 Latent structure from distribution of words

It is believed that meaning of a word can be inferred from the company it keeps, which shows that the semantics of words are largely dependent on the contexts of its usage. This has led to the statistical analysis of distribution of words in large corpora to find the semantic representation of documents. Beginning with bi-gram word occurrences to latent semantic analysis (LSA) many representations are generated from the available linguistic corpora. N-

gram probability estimation from large corpora is a lexical representation, which captures the probabilities of lexical co-occurrence of words and makes useful language model.

LSA analysis (also referred as Later semantic Indexing) of large corpora tries to capture the hidden context or topics from the co-occurrence of words in a document[56,10]. It is useful in inferring relation between words that may not occur together in any of the sentences. In LSA a document is represented as word-sentence matrix. It is a topic modelling technique that hypothesise that document is generated from topics and topics are made up of words. The *term-sentence matrix* of document is decomposed into *term-vectors*, *diagonal matrix* and *sentence vector* using singular value decomposition (SVD) as shown in Equation (2.1).

$$\textit{Term-sentence matrix} = U \sum V^{T} \tag{2.1}$$

Diagonal values in $\sum$ correspond to the topics of the document. Their values indicate their importance value in the document. Although there may not be a clear name of topic, but multiplying the singular left *term-vectors* U with the diagonal matrix $\sum$ gives the relevance score of each term/word for that topic. Similarly coverage of each topic in sentence is approximated by multiplying singular right vector V with matrix $\sum$. This representation is able to identify the hidden semantic relations between words that may not occur together in the document and has been used in many NLP processes including summarisation. Later version of this representation are generated using generative probabilistic models known as Probabilistic Latent Semantic Indexing, which later inspired the development of Bayesian topic model known as *Latent Dirichlet allocation* (LDA)[57]. Although widely used for thematic representation of documents and to analyse hidden structures present in them, these models are still distribution models which are not suitable for generation of proper linguistic sentences and are also not easily human interpretable.

2.3.2 Discourse structure tree

A document is made up of sentences and clauses, which are interrelated to each other. This relation is also known as coherence structure of document. Discourse theories have been postulated to capture this structure and to generate discourse parse trees of document from these discourse relations. Most popular theory of discourse is *Rhetorical structure analysis theory* (RST) [58]. Originally proposed RST theory had 25 discourse relations to link the discourse units of document into a hierarchical parse tree. The discourse units are fragments of original sentences, which are identified by applying syntactic and semantic clues. Discourse relations that link them are *elaboration, cause-effect, contrast* etc. Each discourse relation connects a central unit *nuclei* and dependent unit *satellite*.

Automatic discourse parsers includes rule based parsers where rules are matched with the text to identify discourse units and relations[59–61]. In addition to syntactic information these rules utilises discourse clues in text (i.e. *because, thus*) which are not explicit in many cases to detect accurate relations. Popular discourse parser SPADE trained on probabilistic models has been effective at sentence level discourse detection [62]. Other supervised discourse parsers based on support vector machine (SVM) classifier and decision tree classifier have achieved better performance ratios[63]. Latest state of the art discourse parser for complete text is based on probabilistic discriminative parsing models which utilises inter sentential relations and intra sentential relation[64]. Significant progress has been made is automatic discourse segmentation, but in the case of discourse relation identification still the automatic discourse parsers achieve relatively very low precision than human annotated parse and thus it limits the usage of discourse trees in natural language processing.

2.3.3 Semantic graph

Representation of document's meaning in first order logic by capturing the deep semantic relations between text units comes under semantic graph representation. This representation is capable of capturing information about instances and attribute which is not the case with pure parsing or distributional models. Text units may vary and can be anything such as word, phrase, clause, *subject-verb-object* (SVO) triple or complete sentence. Semantic relations in these graphs are of many types beginning with similarity between text units based on word content similarity or ontological relations such as synonymy. Concept graph is one kind of semantic graph where nodes are words and edges are the conceptual relations between them[65]. A knowledgebase ConceptNet5 has been annotated by group of volunteers to represent the conceptual relations of different words i.e. (*saxophone UsedFor jazz*). Automated concept graph generation from ontologies has been researched to facilitate the search and inferences of hidden relations in the text[66]. Another kind of semantic graph – *logical triple based representation* extracts the main action verb, agent and receiver units from given text and connects these units in graphical structure shown in Figure 2.1 [67,68]. A new semantic graph- *abstract meaning representation (*AMR*) graph* has been proposed by linguistic research community. AMR graph represents the sentences in rooted directed edge-labelled leaf-labelled graphs. It is similar to logical triple representation but has a large set of meaningful relations including relations from prop-bank framesets (arg0, arg1, etc.), semantic relations (*time, location, manner, destination* etc.) and ways to handle negation and modality in terms of polarity and concepts. An example AMR graph is shown in Figure 2.2. A large corpus of annotated sentences in AMR format has been developed to promote work in statistical analysis in semantic graph generation[69]. This has led to development of initial AMR parser and many new works are expected to follow this[70–73]. All of semantic graph

representations are one-step up the parsing and ontology representation and can be enhanced by combining many representations. For example the triple based graph utilises dependency tree parsing and ontological information at the same time and makes it a better reader perceptible representation.

Semantic graphs have been developed at isolated level and no agreed standard parsing software existed until recently the AMR standard came into existence. After recent availability of the corpus *Sembank* in AMR representation few parsers have been developed and utilised in NLP applications including summarisation[74].

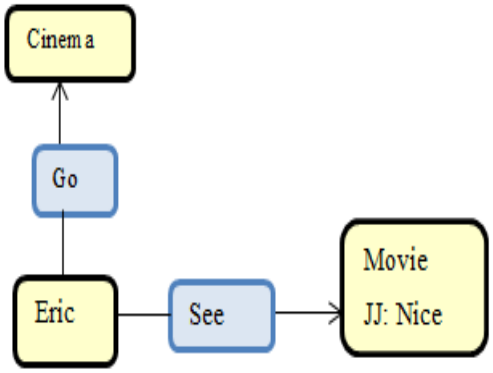

Figure 2.1: Semantic graph.

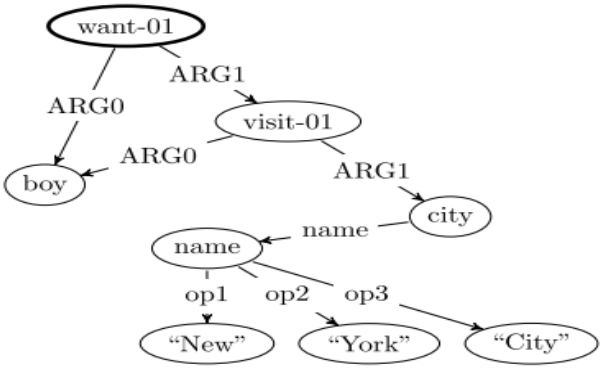

Figure 2.2: AMR graph.

2.3.4 Semantic frame

Semantic frames (SFs) represent textual information in a tabular format for each entity or relation. It is close to representation of information from Object oriented design principle. It has a prefixed template to fill the slots of concept frames from the given text [75,76]. *FrameNet* is the largest annotated knowledgebase for semantic frame data. In FrameNet many words can map to one semantic frame, i.e. SF *Activity_start* that inherits from other SF *process_start* has frame elements (FEs) 1. *Activity, 2. Agent, 3. Co_times activity, 4. Manner, 5. Means, 6. Place and 7. Purpose* and the mapped lexical units to it are: *begin, enter, commence, start, initiate, launch, set_about* etc. Values of FEs can contain reference to other SFs and thus it creates a connected graph of SFs. Frame-semantic parsers have been developed from integrating statistical models derived from different FrameNet versions along with other training data. Recent state of the art semi-supervised parser is trained on FrameNet version 1.5 [77]. There have been attempts at developing unsupervised frame semantic parser which utilised semantic role labelling for prediction of frame and argument roles, but it couldn't achieve much success to identify role names for predicate arguments[78,79].

All of the above described semantic representations combined with syntactic representations has been utilised in automatic text summarisation task. Our research aim is to explore and develop semantic representation of unseen complete documents and summary generation from it. We here review the summarisation approaches with primary focus on the approaches that works through a semantic representation.

2.4 Summarisation methods

Automatic text summarisation has been researched through a mix of strategies for achieving the improved meaningful summaries. Extractive approaches mostly focus on developing efficient rankers of sentences, whereas less researched abstractive summarisation area involves mixed approaches for sentence generation from ranked information. A clear division of summarisation approaches in labelled categories is difficult due to ad hoc mix of strategies. Here in this review we will analyse the summarisation approaches from their central dominant text analysis methods and type of summaries generated.

2.4.1 Extractive summarisation based on sentence level features

These summarisation approaches determine the relevance of information to be included in summary from the statistics of words in corpus and from the sentence features such as position of sentence in the document, keyword matching etc. Here the central belief is that frequency of a word or phrase indicates its importance in the document. Sentences that come in initial positions are considered of primary importance, which holds true for generally experimented news domain data. Generally, words that are most prevalent in the document are taken as *keywords* and summaries are generated from these keywords. Most commonly used *term frequency-inverse document frequency* (*tf-idf*) score of words is computed by its frequency in the document and its inverse frequency in all available document of the particular domain. Latter part makes sure that very common words particularly stop words (i.e. *in, the, a*) are not given higher scores [80,81]. Normalised average of *tf-idf* scores of all words gives the final sentence score in extractive summarisation in basic statistical approach.

These kind of summaries which are not generated from considering the inter document or intra document relations often become victim of redundant information by incorporating too much of high scoring repeated information. Maximal marginal relevance(MMR) approach was integrated with statistical analysis to limit the redundancy of summaries [82]. In MMR approach sentence inclusion in summary is conditioned on its similarity to already included summary sentences. This approach benefitted multi-document summarisation more as the similar content gets repeated in multiple documents. Evaluation of summariser in SUMMAC conference [24] indicated good performance of MMR approach in summarisation.

Statistical methods are useful in theme identification and clustering of sentences around the themes. Themes are the central event or topic discussed in the document. After identification of themes sentences are grouped into cluster based on themes and most representative sentence from each cluster gets included in the final summary[83]. Popular multilingual summariser *MEAD* generates summary from selecting sentences close to centroid of clusters identified from not only one document but from whole corpus for multi-document summarisation[84].

With availability of n-gram probabilistic models more lexical features were integrated into the statistical approaches. Also syntactic features such as part of speech(POS) tags, phrasal grammar connective information has been useful in removing unimportant part of sentences such as prepositional phrases or adjuncts[85] or in identifying important head noun phrases to be included in the summary. Results from the DUC2005 Query based summarisation tasks shows that lexical features improves summarisation by avoiding the pitfalls of statistical based significance index which cannot capture the hidden linguistic relations [86].

2.4.2 Extractive summarisation from lexical connections

Sentence level features are limited to relations within sentences. This makes sentences an independent entity in the document, which is against the principle of sentence coupling for information sharing and cohesion. It prompted researchers to explore the relations within the complete document for extracting summary sentences. It began with realising the connection between different words of the document in form of lexical chains. "*Lexical chains provide a representation of the lexical cohesive structure of the text"* [42]. Lexical chains are built from noun words because nouns are considered the most informative part of sentence compared to other syntactic categories. A long chain of nouns is constructed incrementally by adding new similar words to the existing chains by using semantic similarity measures based on ontology such as *WordNet*. Strength of a lexical chain is approximated by the frequency count of occurrences of its member words in the document and a homogeneity index. Significance of a sentence is calculated from its words/phrases, which are part of identified lexical chains. Words, which are part of strong lexical chains, impart high scores to the sentences. In few approaches first sentence containing the chain members in descending order of their strength are selected to be included in the summary. Some approaches differentiate between the individual scores of member words of lexical chains based on their similarity to the topic words [87,88].

Other than building lexical chains the more effective way of exploring the coherent relations in the document has been by projecting it into a graphical structure. In the summarisation methods, which have scored higher in DUC-2002 summarisation task, the text was analysed by graphical centrality rather than centroid methods. In top performing *TextRank* and *LexRank* summarisers every sentence is mapped to a unique vertex of the graph and the edges connects the sentences which have cosine similarity value above a threshold level[89,90].

These graph representation are ranked using graph ranking methods such as degree centrality, *PageRank* or *Hyperlink-Induced Topic Search* (HITS) to give centrality score to each sentence for extractive summary generation[91]. *iSpreadrank* a multiple document summariser exploits the spreading activation phenomena of social network analysis on the sentence similarity graph by spreading the sentence level feature scores to connected sentences to evaluate the importance scores of sentences[92]. Word Graphs which are based on co-occurrence of words in the sentence or their syntactic relations in the sentence has been utilised along with other semantic and syntactic features for extractive and opinion summarisation[93,94]. These kinds of graphs do not require much language specific linguistic analysis. *UnifiedRank* summariser[95,96] has achieved state of the art performance on DUC-2002 summarisation corpus by integrating single document and multiple-document summarisation tasks into one unified graph framework. It exploits the cross document relations for single document summarisation by incorporating this neighbourhood information into the graph ranking method for extractive summarisation.

2.4.3 Machine learning based extractive methods

From the annotated summarisation corpus a number of supervised and semi supervised extractive summarisation approaches have emerged based on Hidden Markov model(HMM) classifiers, Bayesian classifier and SVM classifier. A Machine Learning (ML) approach towards text summarisation can be envisaged if we have a collection of documents and their corresponding reference extractive summaries. A trainable summarizer can be obtained by the application of a classical (trainable) machine learning algorithm in the collection of documents and its summaries. In this case the sentences of each document are modeled as vectors of features extracted from the text. The summarization task can be seen as a two-class classification problem, where a sentence is labeled as “correct” if it belongs to the extractive

reference summary, or as “incorrect” otherwise. The trainable summarizer is expected to “learn” the patterns which lead to the summaries, by identifying relevant feature values which are most correlated with the classes “correct” or “incorrect”. When a new document is given to the system, the “learned” patterns are used to classify each sentence of that document into either a “correct” or “incorrect” sentence, producing an extractive summary. A crucial issue in this framework is how to obtain the relevant set of features.

HMM based summariser[97] and SVM based summariser [98] were ranked among top 5 performers in the original DUC-2002 summarisation task. Summarisation methods trained on Bayesian classifier determine the probability of a sentence to be a probable summary sentence from the conditional probability of certain features present in it (i.e. position of sentence, *cue words*) [99,100]. According to Bayes’ theorem the probability of a sentence **s** to be part of summary **S** can be calculated by Equation (2.2) where it contains the features F_1, $F_2 \ldots F_n$

$$P(s \in S|F_1, F_2, \ldots . F_n) = \frac{\prod_{i=1}^{n} P(F_i|s \in S)P(s \in S)}{\prod_{i=1}^{n} P(F_i)} \quad (2.2)$$

Probability of feature F_i in the sentence s if it belongs to summary $P(F_i|s \in S)$ and independent probability of Feature F_i $P(F_i)$ can be calculated from the training corpus and $P(s \in S)$ is constant.

Textual data contains huge feature sets and thus supervised SVM classifiers approaches are more popular in information retrieval due to its capability to handle high dimensional data. Summarisation model trained on structural SVM which enforces the diversity and coverage constraints on generated summaries has proved effective on DUC2001 summarisation corpus[101]. SVM classifier trained on ontological semantic features perform better than baseline *tf-idf* score and position based summariser[102]. The heart of SVM classification is kernel method that makes comparison of high dimensional data as simple as dot product

calculation. Kernel methods provide easy ways to convert the parse trees or graph structures of textual information to feature vectors. Various kernel methods have been developed based on syntax parse tree, dependency parse structure and ontological features for NLP tasks of relation extraction, sentence alignment, and paraphrase detection. These Kernel methods have been utilised for sentence extraction in query summarisation. Initially Collins & Duffy [103] introduced *Convolution kernels* for the tasks in Natural language processing. Lodhi et al. [104] developed a string kernel (SK) that is based on the computation of common character subsequence shared between two strings. To reduce the computation cost Cancedda et al. [105] proposed a modified string kernel that works on the subsequence of words instead of characters which is called *word sequence kernel*. To use more semantic features of text for kernel methods new kernels based on part of speech (POS) tag sequence were designed [106].

Chali Y. et al. [107] used syntactic and semantic tree kernel for their multi-document summarisation task and observed better ROUGE score for the experiments. Marcu&Daumé [41] developed an auto text summarizer based on *tree position kernel* for their participation in single document summarisation challenge of DUC-2004. Tree position kernel is based on Rhetoric discourse structure trees of the document and parse trees of individual sentences within the discourse tree. Although the system did not score well in DUC ranking, it did provide good insight into combining two document representation structures for single document summarisation. One more variant of tree kernel is *Dependency tree kernel*, which determines the similarity of two sentences by calculating the common paths between dependency trees of those sentences. This kernel has been used for relation extraction between entities of a sentence in a single document [108,109]. Most of the kernel approaches has been utilized in query-based summarisation, but the research on the impact of linguistic kernels on single document and multiple document summarisation is very limited.

In unsupervised machine learning methods clustering is used to select representative sentences from groups of related sentences. *Columbia Newsblaster system* is an online crawling and summarisation system for news articles based on clustering techniques[110]. In multi-document summarisation clustering techniques have shown better performance than in single document summarisation due to high similarity within the data[111,112].

Neural network based summarisation[113–115] has been done for extractive summarisation and generating highlights of the document. With the new deep learning paradigm which is a neural network based NLP exploration we can expect advancement in its usage in text summarisation[116].

2.4.4 Semi-abstractive summarisation

Previously described summarisation works are purely extractive in nature and although has achieved good performance in summarisation tasks, but suffers low recall in system generated summaries compared to human written summaries[101]. In this Section, we discuss the approaches towards abstractive summarisation, which comprises text-to-text generation methods of compression and fusion.

2.4.4.1 Summarisation from sentence compression

Humans form summary by taking short segments from original sentences instead of using complete sentence. This approach has been explored in summarisation by integrating the NLP task sentence compression.

Integration of sentence compression for summarisation began with Knight&Marcu's noisy channel framework to learn synchronous context free grammar (SCFG) rules for simultaneous generation of compressed sentences and original long sentences[117] together

from the probabilistic language models. This approach assumes that original sentence is a short string, which added with noisy terms forms the longer sentences. Probabilistic model was trained on ziff-Davis corpus, which had 1067 sentence pairs taken from news articles. The probability is estimated from tree generation probability of short trees from long parse trees and word-bigram probability. A decision tree based compressor is also evaluated with noisy channel approach trained on same ziff-Davis corpus with C4.5 algorithm. Noisy channel approach has been found to be more accurate on predicting unseen compressions, while decision tree based compressor is faster in performance.

Galley&Mckeown[118] proposed head driven markovization of SCFG rules based on earlier noisy channel approach and gained better grammatical compressions. Fig. 2.3 shows the parse tree where every syntactic phrase has its lexical head added to it in brackets. Usually in earlier sentence compression approaches preposition words were removed considering them as additional information, but due to added lexical head information now phrases are removed by a semantic approach. In Figure 2.3 prepositional phrase (*PP)* with a head word "*from*" exists in a verb phrase (*VP)* with head word "*fell*" and taking into account the verb argument information we can decide that it's a complement of verb "*fell*" but *PP* with a head word "*because*" is an adjunct. Thus complement can be saved whereas adjunct can be removed during compression.

Following the synchronous grammar's efficiency in generating tree to tree writing rules, Cohn&Lapata [119] has used *Synchronous Tree substitution grammar(STSG)* for generating compressed trees. STSG and SCFG both have grammar rules of type $\langle X, Y\rangle \rightarrow \langle \alpha, \gamma, \sim\rangle$ where X and Y are aligned non-terminal nodes in source and target parse trees. α and γ are the derived subtrees from non-terminals X and Y. SCFG limits the depth of the derived trees α, γ to be 1, whereas STSG allows depth of derived trees to be more than 1.

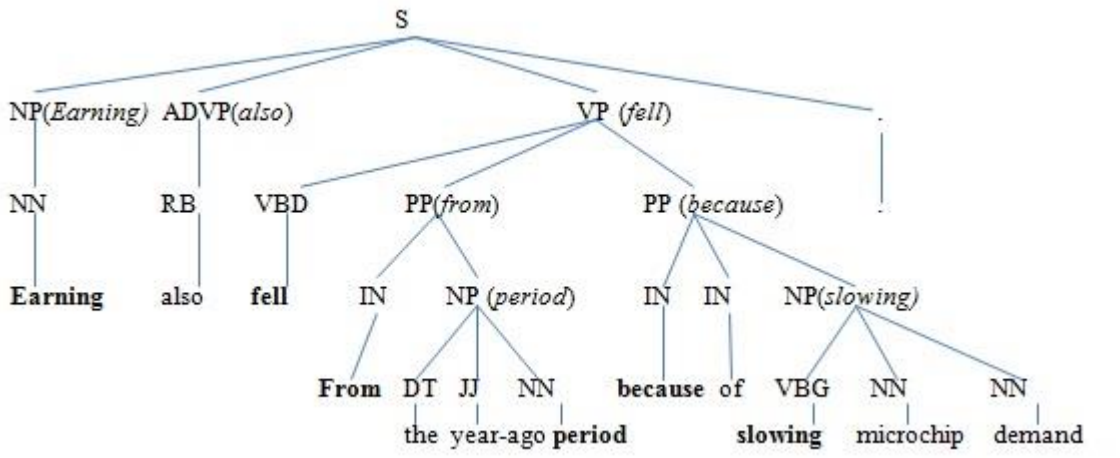

Figure 2.3: Parse tree with lexical heads of syntactic phrases shown in brackets.

In this way, STSG rules allow more than one level of substitution or pruning of the tree. A result of their research indicates that this approach improves compression because it allows reordering of nodes whereas SCFG operations are limited to deletion of nodes. In addition, this approach can be used for other tree rewriting NLP tasks.

ILP solutions [120] for supervised and unsupervised compressions were given which searches for optimal compression by putting global constraints of sentence length, minimum one predicate and correct grammar on the output compressions unlike previous approaches which were visualising compression as a local problem with considering only features of adjacent words/phrases.

To decide the efficacy of compression to produce abstractive summaries experiments on sentence compression and extractive summarisation is performed using ILP based sentence compressions, lexicalised markovian grammar based sentence compression and human authored sentence compressions[13]. Experiments have shown the recall improvement of compression-extraction approaches over pure extractive approaches, but at the same time a big difference from human written summaries points towards the need for language generation.

With the establishment of discourse theory document level compression are formulated as ILP problem with discourse constraints as linear inequalities. This approach has helped in taking compression decision at document level[121]. Jointly performing extraction task and compression on bi-gram features to control redundancy and subtree features for compression by cutting plane algorithm improved performance over extractive baseline[122]. Instead of runtime costly joint extraction-compression a pipeline approach for document summarisation using guided sentence compression was proposed by Li et al. [123]. To speed up the runtime costly joint extractive compressive approaches faster dual decoding algorithms are implemented using dual decomposition [124] and using max/min flow cut in graphical representation of text, which gave comparable results with 100x speedup[125]. Sentence compression results are a step into abstractive summarisation but with a trade-off of slow runtime and lack of constraints to put the structural information into the ILP based declarative solution of word deletion for compression.

2.4.4.2 Summarisation from sentence fusion

Sentence fusion is a NLP task utilised for abstractive style of summarisation by finding common information from similar sentences also called intersection fusion or by combining information from different sentences into one sentence known as union fusion. Barzilay et al. [45] proposed fusion for multi-document news summarisation. It was an improvement over centroid based methods by first clustering similar sentences using a *simfinder* tool into different clusters and then merging the cluster of similar sentences into a fused sentence to form the representative sentence of each cluster. In fusion approaches dependency parse tree of sentences are utilised as common structure to align similar sentences[126]. Filipova & strube[127] proposed ILP based solution for sentence fusion which connects words in dependency trees of similar sentences based on syntactic and semantic similarity and then

from the resulting graph find the optimal merged sentence based on a language model. This approach was more focussed on reducing grammatical errors arising due to earlier fusion approaches. Supervised approaches were not much successful for fusion due to lack of big datasets for training. McKeown et al. proposed a methodology to generate fusion corpus and constructed a corpus of union and intersection fusion of similar sentences [128]. Fusion method, although more common in multiple document summarisation, has been considered for single document summarisation by Elsner & Santhanam [129]. In their approach, a joint optimization problem of sentence merging and sentence alignment was formulated to join disparate contiguous sentences that have information about similar entity or event.

In last few years more work on constructing improved fusion dataset for supervised fusion inferencing has been done by Kapil Thadani et al. This dataset was formed by utilising the manually annotated sentences in the DUC conference for summary evaluations, primarily for pyramid evaluation measure. Their dataset does not suffer from the earlier dataset problem of annotator induces errors. Contrast to earlier single structure based approaches such as dependency parse alignment used in fusion methods they have utilised multi- structures i.e. bi-gram and dependency parse for sentence alignment, merging and generation. Bi-gram structural constraints have been used to produce resulting tree with no cycle between nodes and to put constraints on beginning and ending of resulting tree to be probable beginning and ending of a correct sentence[36,130]. Fusion has been also researched with new terms as multi-sentence compression and sentence enhancement[131,132]. Fusion is the semi abstractive approach which brings summarisation closer to forming new sentences by fusing information together from many sentences. It is more close to human way of abstract generation.

2.4.5 Summarisation from deep rich semantic relations

Although summarisation approaches have followed mixed strategies of extraction-compression, fusion, graphical ranking and discourse based classification using syntactic, sentential and semantic features as discussed in previous approaches there has been a lack of deep semantic analysis to gain understanding of the text document. For a fully abstractive approach deep semantic analysis of document is required for constructing intermediate representation of document which paves the way for concept identification and generation of new text[133,134]. In this Section, we will discuss the summarisation strategies, which are focussed on analysing document semantically before applying heuristics to generate extractive or semi abstractive summaries.

2.4.5.1 Summarisation from latent semantic analysis

The semantic representation of document from the dimensional representation of terms LSA has been first used by Gong and Liu for text summarisation[135]. They selected the sentences, which has largest index values for the top K right singular vectors after the decomposition of *term-sentence* matrix representation of document by Singular Value Decomposition. Similar approaches were used for query focussed summarisation[136]. This LSA base summarisation approach was improved by Steinberger et al. [137] by incorporating semantic information from anaphora resolution and by selecting topics based on summary length. It improved the dangling references issues of LSA based extractive summarisation. Probabilistic latent semantic analysis (PLSA) based on generative models is also utilised in summarisation and found to be better in detecting topics compared to pure LSA[138]. SVD based LSA approaches suffers from the impact of negative values in the singular vector because negative values are not interpretable from textual analysis perspective. Another

dimensional reduction method Non-negative matrix factorization (NMF) has been utilised to remove the pitfall of SVD for latent semantic analysis. NMF based LSA provides non-negative values in the singular vectors for features and topics. Experimental results has shown that NMF based summarisation helps in selecting more meaningful summary sentences[139–142]. There has been some research done on LSA based abstractive summarisation[143]. Abstractive LSA summariser first extracts sentences using pure LSA approach, then reduces the terms in top extracted sentences to only top terms identified from multiplying left singular term vector with diagonal matrix of identified topics. Later a noisy channel approach trained on translation model is used to reconstruct complete sentences from compressed sentences. Mostly LSA has been utilised in extractive summarisation because this kind of representation lacks the labelled relations between words for new sentence generation.

2.4.5.2 Summarisation from semantic graph

As discussed in Section 2.3.3 *semantic graph* brings out the inherent relations in the text document in a graphical form which is easy to be analysed from graph analysis methods. In one kind of semantic graph document structure is represented as a part of domain ontology. Words in the document of POS type noun are mapped to their lexical representation in the ontology and then connecting network of these words generates the corresponding semantic graph. This kind of graph can be easily processed with ontological relations. Semantic Rank [144] constructs the semantic graph of documents using WordNet ontology and Wikipedia knowledgebase. This graph has relations between terms of document derived from semantic similarities of terms based on WordNet's ontological relations and Wikipedia based relations of recommendation. Later *HITS hub/authority score* of the nodes in this graph is used for ranking of summary sentences. Another method has used WordNet based word

disambiguation method to first identify concepts in the sentences and then form concept graph based on WordNet's hierarchical structure to find salient concepts for sentence extraction[22]. These approaches have shown better performance than based on lexical graphs. In specialised domain of clinical text data domain specific ontology Unified Medical Language System(UMLS) and general purpose ontology WordNet has been combined for semantic graph based text summarisation[145,146]. A recent approach on sentence to Wikipedia concept graph and incremental summarisation on this graph is a new direction for single and multiple document summarisation.[147]

Another semantic graph of type SVO triple representations has been first utilised by Leskovec et al. [148,149] for text summarisation by learning substructures of document summaries from document semantic graphs. This graph as described earlier has both ontological relation as well as deep linguistic relations from text. This graph preserves deeper semantic relations by combining SVO triples based on co-references and pronominal-references. In addition, ontological relations from WordNet ontology are utilised to connect the triplet nodes. Each node is also enhanced by additional information taken from the linguistic analysis such as POS tag, cardinality information, and modifier words.

To extract important triples a graph ranking algorithm is applied to the nodes of the graph. Rank of a node in the graph is calculated according to various features such as count of incoming edges and count of outgoing edges from that node. The POS tag, predicate/argument tags (i.e. subject/object/verb) and node types (i.e. person, place) are included in the feature set. After ranking the nodes, high ranking triples are extracted. Later to identify summary sentences, those sentences are selected which contains the extracted triples. For a better performance the process of generating triples and extracting triples from graph can be designed as a machine learning approach by using SVM classifiers[150]. Experimental results has confirmed that domain specific ontologies improves summarisation

in those domains where a text document contains set of special technical words[151]. Rusu et al. [24] describes that word sense disambiguation can be utilized to generate more succinct semantic graphs. Triples based graph built from predicate-argument structures from dependency parse and semantic role labels are utilised to generate event graph with temporal relations built from temporal argument. Graph kernels are applied on these event graphs to generate the multi document summaries of the events happening in some time period[152] and also for narrative text analysis[153]. In a recent state-of the art multi-document summarisation method deep dependency sub-structures (DDSS) are constructed from dependency parses and important DDSS are identified through ILP for extractive summary generation [154]. It has been shown that structures generate from these kind of deep language analysis are more close to human perception of textual information and it constitutes as basic information units of text.

Some abstractive approaches have been also built on semantic graphs, as these graphs preserves the maximum information content of original document for summary generation. Triples extracted from dependency parse has been used in generating abstractive summaries via template filling such as ‘attack’ category of events[155]. A recent substructure prediction problem for summarisation from the document AMR graphs have been decoded as an ILP problem with constraints imposed for connectivity of the subgraph. Although there has been no work on generation of sentences form summary subgraph. Evaluation was done by converting the summary graphs to bag of words and compared against the reference summaries. This is one of the direction changing approaches, which are expected to inspire more research into abstractive summarisation.

2.5 Evaluation of summaries

Summarisation systems are evaluated from the intrinsic and extrinsic evaluation of the summaries generated by them. In intrinsic evaluation of summary the factors evaluated are grammaticality, informativeness, coherence of summary and readability. Extrinsic evaluation determines the usability of summary as a replacement to original document in further tasks. The determining tasks for extrinsic evaluation can be question/answering, information retrieval. The accuracy of results by using original document and the summary gives usability score of summary. In our research work we have evaluated summaries using intrinsic evaluations by comparing co-selection of summary contents with human written summaries for those documents. Grammaticality of extractive summaries is similar to original text content but grammaticality of abstractive summaries has been evaluated manually.

There are many intrinsic evaluation systems developed over the course by different research groups and conference organisers for comparing system summary with human written reference summary. Among this the standard adopted in Document Understanding Conference is *ROUGE evaluation system*. ROUGE is n-gram comparison metrics to determine recall and precision of summary content. It has been used in our research to evaluate the different extractive and abstractive summarisers. Since most of the evaluation parameters are quite subjective, no evaluation system can assuredly conclude whether one summary is good for the purpose or not. Considering this additional evaluation is done using other systems that are described later. Here we will describe the ROUGE metrics and in short other standard evaluation systems [156].

2.5.1 ROUGE evaluation

To minimize the human efforts involved in summary evaluation process, an evaluation measure ROUGE (*Recall-Oriented Understudy for Gisting Evaluation*) was introduced in DUC-2004 (Document understanding conferences) [157]. It is adaptation of BLEU evaluation system, which has proven effective in automatic evaluation of machine translation. ROUGE is based on n-gram content overlap between automatic summary and human written summary. Before 2005 ROUGE had only recall metrics, but latest ROUGE Version 1.5.5 has recall, precision and F-measure. Complete ROUGE package has various methods for evaluation of summaries. A short description of these methods is given here.

1) ROUGE-N: It calculates recall of common n-grams between automatic summary (A) and many referenced summaries.

$$\text{ROUGE-N} = \frac{\sum_{S \in reference\ summaries} Count\ of\ common\ n-gram\ between\ S\ and\ A}{\sum_{S \in reference\ summaries} Count\ of\ n-gram\ in\ S} \quad (2.3)$$

Rouge-n score is generally calculated for unigram, bigram, 3-gram match. A summary that has more common n-grams with all the referenced summaries gets a higher ROUGE-N score and thus this measure evaluates which summary is more agreed upon by all human judges. This measure will best avoid human disagreement if we have many referenced summaries written by different authors. ROUGE-1 has been found to be close to human evaluation.

2) ROUGE-L: A sentence is considered as a sequence of words and Rouge-L score determines the structural similarity between sentences of summaries by the longest common subsequence (LCS) shared between them. It works on the belief that similarity of summaries is dependent on the similarity of its sentences. Gaps are allowed when counting LCS, so it looks for in-sequence but not consecutive matches. High ROUGE-L scores indicate the sentence-structure level similarity between system summary and referenced summary.

ROUGE-L score between automatic summary sentence A_i and reference summary S_i of length m is computed by following formula.

$$ROUGE - L = \frac{LCS(A_i, S_i)}{m} \quad (2.4)$$

3) ROUGE-W: It is rouge score of weighted longer common sequence shared between summary sentences. It gives more weightage to consecutive sequence matches. At each matching index weight is calculated from previous longest match at last matching index and a function to penalise the gaps. Length of the consecutive sequence matches is utilized to calculate the weights.

4) ROUGE-S: It measures the overlap of skip-bigram between automatic summary and references summary. Skip bigram are bi-grams with any gap between the coupled words. Although to avoid the inclusion of stop-words into skip bigram a limit is imposed on the distance between words. One other variation of this method is ROUGE-SU, in which common unigrams (1 word) count are also included for final ROUGE-S score.

Significance of co-relation tests between Rouge evaluation and human evaluation on DUC data shows that ROUGE evaluation works well on single document summarisation and short summaries and high levels of agreement with human evaluation are observed. For multi-document summarisation evaluation, ROUGE results were not comparable with human evaluation but exclusion of *stopwords* from summaries improved the co-relation. ROUGE-2, ROUGE-L, ROUGE-W, and ROUGE-S are the best performing measures in single document summarization. ROUGE tool is a part of standard evaluation system for DUC conferences and utilised in our research work.

2.5.2 Other intrinsic evaluation system

Other than standard ROUGE metrics, there are other evaluation systems proposed for intrinsic evaluation of system generated summaries. Pyramid method was proposed by Nenkova et al. [158] in 2004 for content evaluation of summaries. The basic difference from Rouge method was its methodology to handle the differences in content selection by human summarisers for gold standard reference summaries. Pyramid methods evaluate the summaries not based on sentences or n-grams but based on meaningful *summary content units* (SCUs) which may be expressed in different words in referenced summaries. These SCUs are given weights according to the count of summaries in which they appear and a pyramid of SCUs is built for each document set. Later system summaries are scored according to the SCUs in the pyramid structure. System summaries are also manually annotated for the presence of SCUs. Pyramid method has been utilised as a supplement evaluation in DUC conference 2005-2007[159,160].

Functional assessment of pyramid method on DUC conference data has shown its effectiveness in evaluating system summaries for content similarity with reference summaries[160]. Pyramid method seems to be robust to inter annotator disagreement by giving consistent scores to same system summaries annotated by different peers. The drawback remains in huge manual effort required first to generate SCUs from human authored summaries and then peer SCU expressions from system summaries.

To encourage analysis and development of automatic evaluation measures for summaries a new task has started from 2009 in TAC conference named *automatically evaluating summaries of peers (AESOP)*. Participating work for this task promotes using content of original text document for summary evaluation instead of human written summaries to avoid the manual disagreement completely for annotation and content selection[161–166].

Overall valuable work has been done on the content evaluation of summary where a summary's information content is analysed in terms of content coverage. However, qualitative evaluation is still under development. In addition, most of the evaluation methods require human generated summaries, which is labour intensive. Although there is past data available from DUC conferences, but a complete automatic evaluation method, which is only dependent on the original text document and available domain knowledge will be more desirable.

2.6 Corpora and conferences

There are three major conferences for text summarisation TIPSTER Text Summarization Evaluation (SUMMAC), Document Understanding Conference (DUC) and NTCIR. SUMMAC was organized by U.S. government to evaluate research works in automatic text summarisation field. In year 1997, SUMMAC started with six summarisation systems, as an informal test run of SUMMAC. In 1998 SUMMAC was organised at large scale to evaluate the Summarisation systems for their usefulness in 2 NLP tasks: *relevancy measurement of documents* and *question/answering*. Total 16 systems participated in the evaluation that included recognized Universities and Industry research groups (IBM Watson research centre, British Telecommunications, TextWise LLC, and SRA International). SUMMAC conference has confirmed that text summaries are as useful as original complete text for the relevance measurement task of documents. In addition, it opened the way for evaluating usefulness of automatic summaries to the different NLP tasks. A corpus of 183 scientific documents in xml format along with abstract is made available for summarisation evaluation tasks. This corpus is publically available on SUMMAC website.

NTCIR is another major evaluation workshop for encouraging research in information retrieval field including text summarisation. It was originally started by *Japan Society for Promotion of Science* (JSPS) and *National Centre for Science Information Systems* (NACSIS).Various tasks for intrinsic and extrinsic evaluations of summarizers was organised from year 200- 2004. That includes relevance measurement, evaluation of single and multi-document summaries, linguistic quality evaluation of summary and usefulness evaluations through Question/answering task. Corpus available from the summarisation tasks in 2000-2004 are NTCIR-2 SUMM, NTCIR-2TAO, NTCIR-3 SUMM and NTCIR-4 SUMM. Corpus is available at request with a fee for the non-participating research groups.

In year 2000 U.S. Government initiated a new evaluation program that eventually led to first DUC conference in 2001 [167]. From year 2008, this program has become part of Text analysis conference (TAC). Since its beginning the main goal of DUC has been to evaluate summaries according to the linguistic quality and content coverage. As described in Section 2.5 various evaluation methods were applied in DUC to check the quality of summary in terms of grammar, connectedness, and referential clarity. Automatic summaries were compared with human generated summaries for evaluation of content coverage. DUC-2001 and DUC-2002 conferences included generic summary evaluation tasks. Since generic summaries are more difficult to evaluate in terms of its coverage, later it was decided to work on focussed summaries. In DUC-2003 and DUC-2004 new tasks topic detection and even detection were included. From 2005 more focussed task of generating ~250 words query base summary from multiple documents was included. It continued in later all DUC conferences with an added task of generating ~100 words update summary from multiple documents.

Corpus of past DUC evaluations tasks in available on the DUC (2001-2007) and TAC websites. License for this data can be obtained without any fee, by sending request to the

maintaining group. Another open source evaluation corpus for summarisation tasks is Columbia University's *Newsblaster* data.

2.8 Limitations of current summarisation approaches

Most of the research in text summarisation has been focussed on extractive summarisation. Sentences are selected according to position in text, words statistics, syntactic importance or semantic connectivity between sentences. These types of summaries will satisfy the grammaticality criteria of ideal summaries and are useful for topic identification, question answering and relevance measurement. The drawback of extractive summarisation is that generated summary will not be coherent, as all the extracted sentences may not logically connect to each other due to dangling references. Another drawback is that summaries will not rank high in terms of content coverage and informativeness due to extracted long sentences with redundant information occupying the limited summary space. It inspires to prioritize research towards abstractive summary generation.

A significant approach in the direction of more coherent summaries is sentence compression. As discussed earlier it tries to reduce every sentence into short length sentences according to the probabilistic grammar rules. Keeping shorter form of every sentence into summary will increase content coverage of summary. However, at the same time it is a supervised learning method and requires huge corpus of training data to measure the probabilities of grammar rules. Rich semantic approaches such as semantic graph, latent semantic indexing gives importance to semantic relations between words and the scoring of sentences is also based on different semantic relation or dependency relation shared between them. Again, as sentences

are extracted as independent units, it will not produce a complete coherent and connected summary.

Fusion approach is a step towards generating abstractive summaries. Its language model makes summaries grammatically correct. To keep the summary coherent and connected only those part of the sentences are added whose root node is already present in the syntax tree of common most sentence chosen as basis tree for fusion [45]. Fusion approach is more suitable to multi-document summarisation, because many documents about same topic/event (i.e. news articles) contain common sentences that share same information content. Fusion of these sentences generates an informative shorter text document. In a single document, information is repeated rarely thus, sentences may not share much common information, but still refer to some common topic or entity. Union Fusion explained in Section 2.4.4.2 is applicable to these sentences in single document summarisation. Considering the difference between the single document and multi-document summarisation, recently there has been comparatively more research progress in multi-document summarisation due to high repetitive content. Generic summarisation of single documents has not been researched very actively in last few years compared to focussed summaries i.e. query based summarisation or topic based summarisation [114]. It is largely due to deeper semantic analysis required for identifying important information from single documents, which is not biased towards keywords for any topic. Deeper semantic analysis of text documents is about understanding the meaning of the content and then processing it. It requires developing a semantic structure, which can assemble the information of textual data into smaller connected units for ranking and generation of new information. This is one of the objectives of our research described in Chapter 1 to design an efficient semantic representation for text documents for applications in text summarisation. To overcome the issues of extractive summaries our second objective set in Chapter 1 is to analyse approaches towards abstractive summarisation and derive

methodology to generate abstractive summaries from semantic representations. In next chapters we will describe our work towards these objectives.

Chapter 3

Effect of clause splitting on sentence alignment

3.1 Introduction

The aim of our research, as described in Chapter 1, is to analyse, construct semantic representations for textual information, and develop automatic summarisation methods to generate abstractive summaries. In the literature review Chapter where past text summarisation methods have been described sentence fusion has emerged as one of the most plausible approaches towards abstractive summary generation. Sentence fusion generates can lead to either semi abstractive summaries or fully abstractive summaries. Its basic mechanism is to align many similar sentences to find most common part of information presnt in all sentences and then enahnace the common information with other omitted parts of sentences. Analysis of the first substask in sentence fusion -*sentence alignment* led to our first research study. Sentence fusion involves aligning many similar sentences to identify common information and uncommon information between them. This sub-problem of aligning sentences is called *sentence alignment*. Sentence alignment is a well-defined task of NLP used in machine translation, question answering (Q/A), entailment check and paraphrase generation [168].

In sentence alignment syntax trees of many similar sentences are aligned to find overlapping information between them. After sentence alignment subtask, next subtasks of sentence fusion includes enhancing the the basic tree found by sentence alignment with additional

information taken from the omitted parts of all sentence parse trees [45]. Sentence alignment task is not only used in sentence fusion summarisation but also in other NLP tasks such as sentence compression and redundancy removal. To shorten originally long sentences into informative smaller sentences, parse trees of compressed sentences and original sentences are aligned for automatic rule learning from training corpora [118]. In few summarisation approaches other than fusion summarisation *sentence alignment* is used as a similarity measure to remove redundant sentences from the summary and to evaluate extractive summarisation by aligning sentences of extracts with human authored summaries [12]. In this chapter we focus on sentence alignment which is common task among many summarisation approaches and then in next chapter we focus on pure abstractive technique-*Sentence fusion*.

From reviewing preceding work before our research study on sentence alignment we saw that initially sentence alignment methods were purely statistical [169]. These maximum likelihood approaches were based on word length and character length in parallel sentences. Alignment work by Wu [170] was first to utilise lexical information and it was the basis of further similar lexical similarity based approaches on sentence alignment[171–173]

Later to introduce grammatical constraints for alignment a new method was incorporated to align syntax trees of bilingual corpora [174]. Similar techniques were introduced to monolingual corpora and improved by inclusion of semantic similarity by Barzilay et al. [175]. This approach converts sentences to their dependency relation tree and then alignment is performed as global alignment of dependency trees. This alignment approach is the most common approach used for fusion summarisation and paraphrasing tasks. Further research in fusion by Marsi&Krahmmer [126], and Fillipova&Strube [127] are also based on similar lines of work to align dependency trees of sentences.

There are other phrase based alignment methods which have been used in natural language inference (i.e. entailment check) but are not suitable for fusion based summarisation due to the missing symmetric features in input sentences for fusion [35,36].

In our research we follow the dependency based alignment procedure due to its wide usages in fusion summarisation. Also the semantic representation of sentences in form of predicate-argument structures of dependency relations made it more suitable for our preliminary research on semantic representation. We propose to improve the performance of this alignment method by splitting sentences into clauses before alignment. Apart from improving time performance this helps in avoiding wrong alignment of non-similar nodes due to high alignment scores of their long subtrees. Splitting sentences into clauses forms smaller dependency trees and thus avoids the latter. In Section 3.2 we describe the original alignment method proposed by Barzilay et al. In Section 3.3 we describe the proposed alignment method in our research. In the next sections we describe the experiments and analysis of results and their effect on alignment using our new method.

3.2 Methodology

The original method of sentence alignment by Barzilay et al. [175] works on the dependency tree structure of input sentences. A dependency parse tree of the source sentence is aligned to the dependency parse tree of the most similar sentence of the documents for fusion of information. To find the common part of the sentences all possible combinations of pair of subtrees between source and target trees are explored. Dynamic programming solutions for this method of alignment improves time performance but still this approach can cause combinatorial explosion if the count of children grows in the source and target dependency trees [176]. This is a probable cause for lengthy sentences of the news domain, which is the

most popular domain for summarisation. Our experiments on dependency tree generation show that in subtrees the count of child nodes increases with the count of words in the original sentence.

Here first we give small description of dependency parsing and Stanford's dependency set which is the standard dependency relationship set used in our research. After that we describe the original and proposed algorithm for sentence alignment.

3.2.1 Dependency parsing

Dependency grammar is one of the syntactic representations of sentences other than phrase structure grammar. It describes the structure of the sentence in terms of dependency relation between the lexical units-words. The dependency relation is an asymmetrical relation between words in sentence, where one word is *governor* or *head* of the relation and the other word is *dependent*. Dependency grammar has gained popularity recently in various NLP tasks due to the predicate-argument structure of dependency relations. Stanford's phrasal structure based dependency parser is the best performing dependency parser and most utilised in academics and Industry based research [24]. Stanford's typed dependency relations are one of the standard relations proposed for English language. There are 50 dependency relations i.e. *nsubj*- nominal subject, *nsubjpass*- passive nominal subject in the original Stanford dependency representation. These dependency relations were later extended for other languages i.e. Chinese, Spanish. To make a uniform annotation standard across languages a Universal dependency(UD) representation has been proposed based on the Stanford dependency relation [177]. The latest version of the Stanford parser gives output in UD representation, however during this study and in subsequent research work presented in the thesis we have used original dependency representation of Stanford. Most common

dependency relations are *nsubj*, *dobj*, *pobj*, *iobj*, *adj* and *adv*. A root node is added to the main verb of the sentence to make a connected tree structure. Figure 3.1 shows a small dependency parse tree of a simple sentence shown below with its dependency relations. This visual graph is generated by using *Jgraph* APIs in our implementation. We see that the main verb "rises" is connected to the root node.

The Sun rises in the east and it sets in the west.

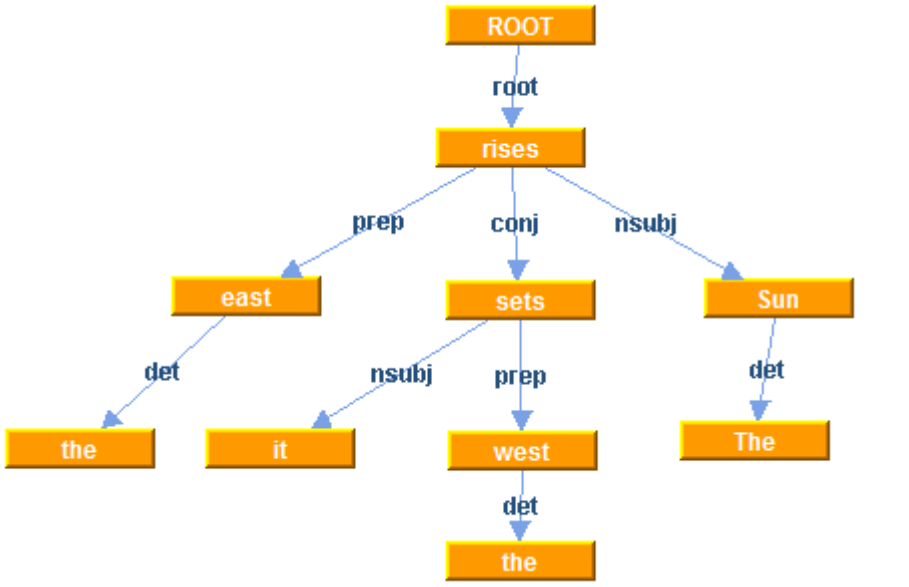

Figure 3.1: Dependency parse tree.

3.2.2 Original sentence alignment algorithm

In the original alignment algorithm the dependency tree of the source sentence is aligned to the dependency tree of the target sentence. Target sentence is chosen to be the shortest sentence among the two sentences to be aligned. The dependency parse tree of this shortest sentence is the target tree. During alignment the following steps are performed to explore all combinations of subtrees for comparison:

(i) Root of first tree gets aligned to root of second tree

(ii) Root of first tree gets aligned to all children of root of second tree

(iii) Root of second tree gets aligned to all children of root of first tree

The best alignment found in the above steps is taken as the final alignment of the sentences. At each pair of the node above, steps are called recursively and that leads to a bottom up alignment approach formulated as below[monika].

For any two trees T_{V_1} and T_{V_2}, which are rooted at nodes V_1 and V_2 and their children are denoted by C(T) then their similarity $Sim(T_{V_1}, T_{V_2})$ is calculated by taking the maximum of the scores calculated in above three steps. The first step is calculated by *NodeCompare(v_1,v_2).*

$$NodeCompare(v_1, v_2) = nodeSimilarity(v_1, v_2) + \max_{m \in M(c(v_1), c(v_2))} \left[\sum_{s_1, s_2 \in m} \left(EdgeSimilarity((v_1, s_1), (v_2, s_2)) + Sim(T_{s_1}, T_{s_2}) \right) \right] \quad (3.1)$$

In (1) *nodeSimilarity(v_1,v_2)* is the similarity score of two nodes based on semantic features or lexical content overlap. The remaining second expression is the maximum of total similarity score of edges of different combination of mappings M(c(v_1),c(v_2)) between children nodes in c(v_1) to children nodes in c(v_2) and best alignment scores of subtrees rooted at these child nodes for each mapping of children nodes. Edge similarity is calculated based on dependency relation between nodes.

Step 2 is $\max_{s \in c(T_1)} Sim(T_s, T_2)$ and step 3 is $\max_{s \in c(T_2)} Sim(T_1, T_s)$. Step 2 and step 3 recursively calls *NodeCompare(v_1,v_2)* based on the root node of one tree and children nodes of other tree. We can see from the above formulation that this approach builds alignment of trees in a bottom-up way. That is we first compute the optimal alignment probabilities of small trees and use them to compute that of the bigger tree by trying different alignment configurations. This procedure is recursive until the optimal alignment probability of the whole tree is

obtained. This is bottom up manner alignment of subtrees[232]. It “may force alignment of two unrelated words if the subtrees they root are largely aligned” [127]. This is one of the recognised issues in this alignment algorithm.

3.2.3 Proposed clause splitting improvement

In the original alignment algorithm explained in the previous Section, we see that during comparison of nodes their children are mapped to each other for all possible permutations of nodes in $c(v_1)$ *to* $c(v_2)$. For trees with a large number of child nodes this mapping may increase exponentially and thus may cause performance issues for further computations on these mapping sets. Other recognised issue with this alignment algorithm is wrong alignment of non-similar words due to bottom-up alignment.

To overcome the performance issue and to avoid the alignment of non-similar words due to bottom-up approach, we have proposed to split the sentence into smaller meaningful clauses and then perform alignment of clauses separately. We postulate that small clauses will generate smaller dependency trees and will reduce the combinatorial increase of mapping sets. Thus this will lead to time performance improvement of sentence alignment algorithm.

Theoretically we observe that after clause splitting each tree pair to compare has a smaller set of child nodes $c(v_1)$ and $c(v_2)$, which decreases the permutation count of $c(v_1)$ and $c(v_2)$ and thus reduces the size of mapping set $M(c(v_1), c(v_2))$. Computation cost of the following operations is reduced after reduction in mapping sets.

$$\max_{m \in M(c(v_1),c(v_2))} \left[\sum_{s_1,s_2 \in m} \begin{pmatrix} EdgeSimilarity((v_1,s_1),(v_2,s_2)) + \\ Sim(T_{s_1}, T_{s_2}) \end{pmatrix} \right] \quad (3.2)$$

Combining the alignments together with clause alignments requires further computations but has comparatively less computation cost than the reduced cost from the operation in Equation (3.2).

Also due to smaller trees the distribution of alignment score is more balanced and avoids aligning of non-similar top nodes due to largely aligned longer subtrees. We also proposed a method to combine the clause alignments to form a complete alignment of two sentences. We here explain the steps performed to implement the proposed method and then describe the experiments done for measuring the impact on performance.

3.2.3.1 Clause splitting

Natural language sentences are made up of clauses. A sentence can be limited to one single clause or multiple clauses. Basic requirement to be a clause is that it should have a predicate. Clauses can be finite clauses, which have a tensed verb and a subject, or non-finite clauses where no tense information is attached to verb and the subject may be missing. To make sense of non-finite clauses we need to look at the main clause attached to it. Clauses are also subdivided according to the role they play in a sentence such as noun clause, verb complement, adverbial or adjectival clause.

Clause splitting identifies the clauses in the sentence. Clause splitting methodologies for English have been developed to identify clause boundaries within a sentence by using the conjunction words or subordinators [178], or machine learning approaches using perceptron [179] and random fields [180]. Clause boundary identification has been difficult due to embedded clauses in complex sentences. We follow a rather simple approach to identify subordinate clauses of type finite only by following the clause level parser tags. We use this simple approach to analyse the initial effect of segmenting sentence into clauses and then

aligning them. Penn Treebank tag set[181] defines clause tags *S* for simple clause. A subordinate clause which starts with subordinator words (i.e. *that, but, if, after, until*) is tagged by *SBAR*. *SBARQ* enclose the clause beginning with question-word (i.e. *when, while*). Stanford syntax parse with accuracy rate of 86.36% have been utilised to generate the parse tree structure of the sentences. Examples describe the clauses enclosed with tags generated from Stanford parser.

I first met him in Japan, where I was spending my holidays.

```
(ROOT
  (S
    (NP (PRP I))
    (ADVP (RB first))
    (VP (VBD met)
      (NP (PRP him))
      (PP (IN in)
        (NP (NNP Japan)))
      (, ,)
      (SBAR
        (WHADVP (WRB where))
        (S
          (NP (PRP I))
          (VP (VBD was)
            (VP (VBG spending)
              (NP (PRP$ my) (NNS holidays)))))))
    (. .)))
```

Edge similarity in Equation (3.1) is calculated from the dependency relation match between nodes. The root of the dependency tree is predicate thus correct clause identification with each clause having a predicate preserves the original dependency structure of the complete sentence in the smaller dependency trees of clauses.

3.2.3.2. Alignment of clauses and scoring function

We follow a similar strategy for scoring clauses as formulated for original sentence alignment. After a sentence has been broken down into its clauses by the clause splitting method the alignment score for a pair of clauses is generated from Equation (3.1). In implementation of Equation (3.1) node similarity of two nodes is calculated by exploring the synonymous relations between them in the WordNet ontology and by their lexical similarity.

In Equation (3.1) *EdgeSimilarity* is straight comparison of dependency relation labels of edges with similar source nodes. After alignment scores for pair of clauses has been computed a combining function is utilised to combine clause alignments to form original sentence alignment scores, which is described in next Section.

3.2.3.3 Combining clause alignments

For every pair of sentence to be aligned source clause set *S* and target clause set *T* are constructed from the clause splitting method. The sentence with the smallest number of clauses is taken as the source sentence.

$S=\{Clause_{S1}, Clause_{S2}, \ldots\ldots\ldots Clause_{Sn}\}$

$T=\{Clause_{T1}, Clause_{T2}, \ldots\ldots\ldots Clause_{Tm}\}$

For each possible pair of mapping from S→T, the alignment score is calculated by Equation (3.1). To combine clause alignments the maximum scoring clause pairs are added one by one to the final alignment pairs. To ensure that every clause appears only once in the final alignment the combining function selects only those clause pairs which do not exist already in the alignment list. Clause alignment scores have to be better than certain threshold score to avoid only stop-word alignment. Threshold score of 200 has been determined from expermentation on different length sentences. Due to our simple clause splitting method which only splits clauses which are clearly subordinate clauses identified with clause tags, the approach generates ambiguities when a clause in one sentence may wrongly be broken into two clauses and partially matches two clauses of the other sentence. Additional issue arises due to structural difference in sentences when some words are not aligned when seeking the best clause alignment pairs. To resolve these ambiguities we acknowledge the need to look at the second best alignment for each clause and we take the aligned words from the second best aligned clause for the non-aligned words of the first best alignment. Pseudocode of the

combining function for clause alignment is shown below. The remaining pseudocode is shown at the end of the Chapter.

```
Clause_Alignment (){
//main function
Initialize ListOfBestAlignment, and ListOfSecondBestAlignment;

//Split the sentence into clauses
ClausesList_source=Split (Sentence1);
ClausesList_target=Split (Sentence2);

/*this step makes the sentence with less number of clauses as the first sentence*/
If (countOf(ClausesList_source)> countOf(ClausesList_target))
Exchange the sentence and clause Lists;

/* For each clause combination sourceClause_i , targetClause _j in ClausesList_source and ClausesList_target get the maximum scoring alignment using original algorithm and add it to ListOfBestAlignment. For further analysis add second maximum scoring alignment to ListOfSecondBestAlignment */
For each clause sourceClause_i in ClausesList_source
{
   Score=0; bestScore=0; secondBestScore =0; Alignment=null;
   For each clause targetClause_j in ClausesList_target
   {
     (score,Alignment)= Sim(sourceClause_i, targetClause_j) //original algorithm
     If score>bestScore
      {
        secondBestScore=bestScore;
        SecondBestAlignment= bestAlignment;
         bestScore=score;
         bestAlignment=Alignment;
      }
    }

  If ListOfBestAlignment already contains best aligned targetClause for some other alignment:
  Compare the scores of new and existing alignment and add higher scoring alignment to list.
  ListOfBestAlignment.add(bestAlignment);
  ListOfSecondBestAlignment.add(SecondBestAlignment);
}

//calculate combined alignment score of sentences
Initialize AlignmentScore=0;
For each alignment in ListOfBestAlignment,
AlignmentScore = AlignmentScore+ alignment.bestScore;

/*If one clause sourceClause_i matches more than one clauses targetClause_j of Clauses sentence_2 then add those aligned nodes from ListOfSecondBestAlignment to ListOfBestAlignment which are not aligned in best Alignment of sourceClause_i.*/
for each alignment (sourceClausei, targetClausej) in ListOfSecondBestAlignment
{
 If secondBestScore>200(threshold value)
 {
   Add previously unaligned nodes from sourceClause_i, targetClause_j to ListOfBestAlignment
```

```
   /*Add normalized secondBestScore to combined AlignmentScore of two sentences*/

    normalized secondBestScore= secondBestScore  ×
    (number of nodes added to best alignment from secondBestAlignment / total number of nodes in secondBestAlignment)
    AlignmentScore= AlignmentScore+  normalizedsecondBestScore;
  }
  Return (ListofBestAlignment,AlignmentScore);
}
}
```

Assessment of the results before and after including aligned nodes from the second best alignment shows reduction in alignment loss. We illustrate the proposed alignment process on the following sentences 1 and 2. The corresponding dependency parse trees Tree A and Tree B are shown in Figure 3.2 and Figure 3.3. These dependency parse trees are generated from Stanford's dependency parser. In previous section 3.2.1 dependency parsing is explained in detail. Generally depeendecy parsers identifies predicates and relations from syntactic parsing of the sentences. Sentences for the illustration of proposed alignemnt process are:

1: After the taping, she said, Jackson left and they did not see him again until one day his associates arranged a private jet flight to a Miami resort where the pop star was waiting for them with a large group of people.

2: After the taping, she said, Jackson left and they did not see him again until unusual events began to happen

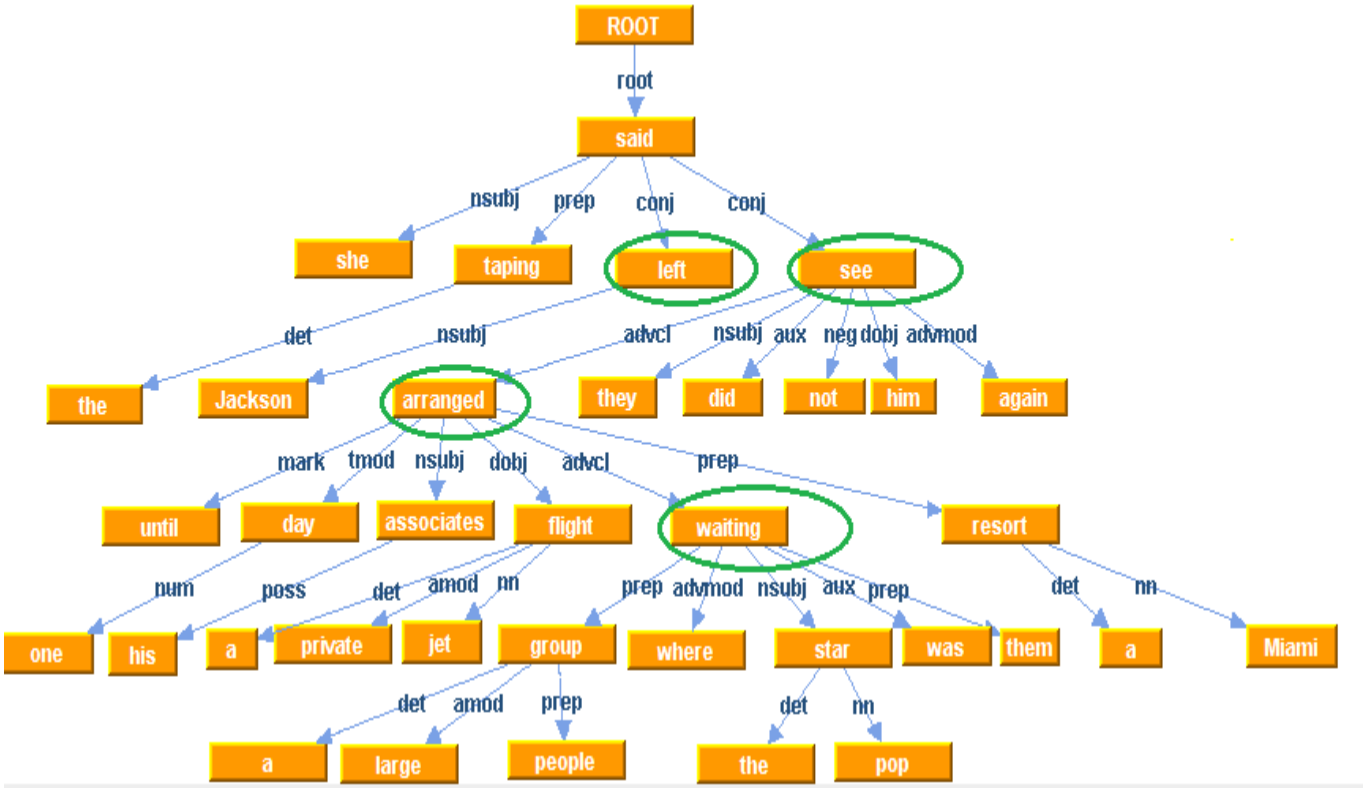

Figure 3.2: Dependency tree A for sentence 1.

Following the original approach of sentence alignment the alignment of dependency tree A and B will lead to comparison of children of root node “said” in tree A. ([She, taping, left, see]) to children of root node “said” in Tree B ([She, taping, left, see]). The mapping at this subtree level will cost similarity calculation of 96 pairs. (Number of nodes to match × number of combination of nodes=4× ${}^{4}C_{4}$ =4*24=96).

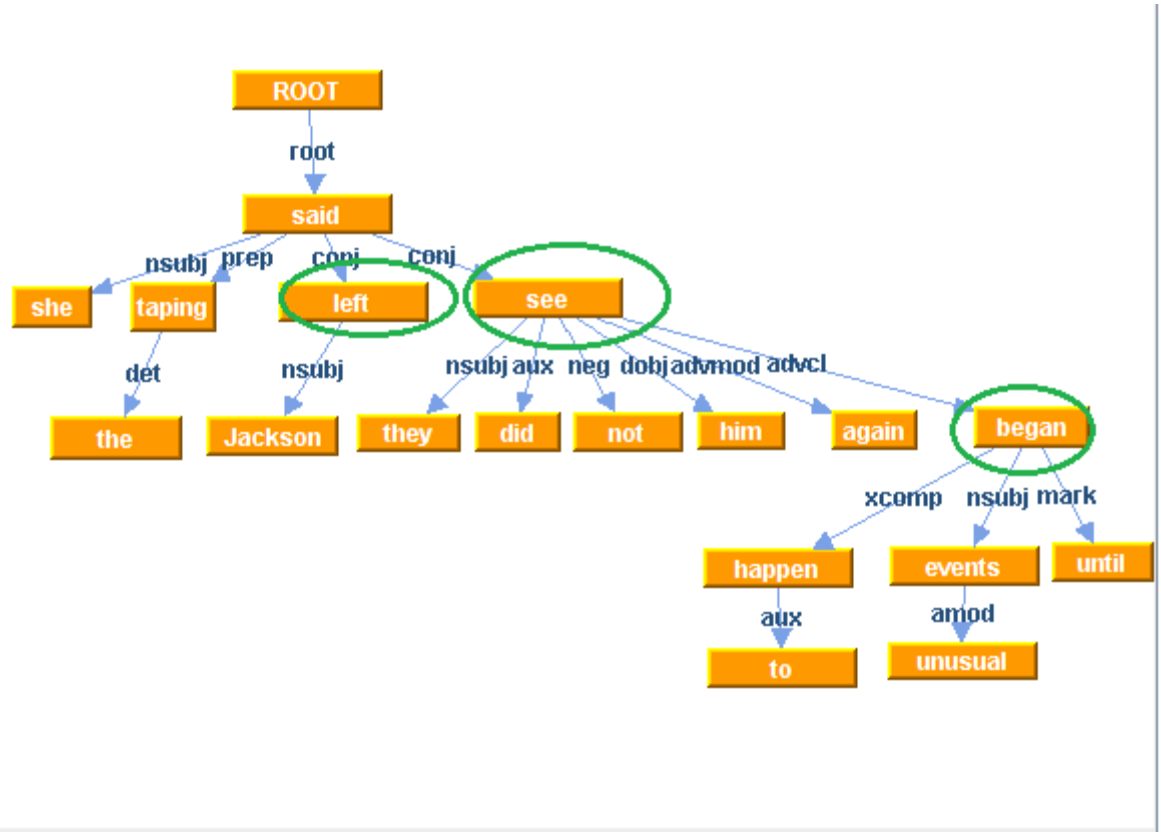

Figure 3.3: dependency tree B of sentence 2.

The proposed new approach of sentence alignment using clause splitting divides the sentences 1 and 2 into following set of clauses.

Source clause set:

1.1: After the taping, she said

1.2: Jackson left

1.3: and they did not see him again

1.4: until one day his associates arranged a private jet flight to a Miami resort

1.5: where the pop star was waiting for them with a large group of people

Target clause set:

2.1: After the taping, she said

2.2: Jackson left

2.3: and they did not see him again

2.4: until unusual events began to happen

In the dependency tree A and B shown in Figure 3.2 and 3.3 the circled nodes are roots of clause trees. After splitting sentences into clauses, independent dependency trees form rooted at these circled nodes. This leaves two immediate children “She” and “Taping” in both trees rooted at “Said”. In Table 3.1 we have shown the similarity computations at immediate children level of node “said” between all alignments of above shown clauses. A total of 16 similarity checks are there. We ignore the comparisons which are similar to step 2 alignment of the node with other tree’s child nodes. These comparisons do not count as additional similarity checks because they are part of original algorithm. This example shows the reduction of similarity checks after clause splitting. In next Section we see the experiments done for a large corpus and the results.

Table 3.1: Mappings of nodes for similarity check.

	2.1	**2.2**	**2.3**	**2.4**
1.1	$She_A \rightarrow taping_B$, $taping_A \rightarrow She_B$, $She_A \rightarrow She_B$, $taping_A \rightarrow taping_B$	$She_A \rightarrow Left_B$ $taping_A \rightarrow Left_B$	$She_A \rightarrow See_B$ $taping_A \rightarrow See_B$	N
1.2	$Left_A \rightarrow She_B$, $Left_A \rightarrow taping_B$,	$Left_A \rightarrow Left_B$	$Left_A \rightarrow See_B$	N
1.3	$See_A \rightarrow She_B$ $See_A \rightarrow taping_B$	$SeeA \rightarrow Left_B$	$See_A \rightarrow See_B$	
1.4	N	N	N	N
1.5	N	N	N	N

Final best alignment

[court37--Attorney40, one20--one15, that15--that10, complaint34--complaint42, Madoff1--Madoff1, big21--big16, it16--it11, just19--just14, ROOT0--ROOT0, employees4--employees4, the3--his6, all18--all13, lie22--lie17, the33--the38, is17--is12, told2--told2]

Alignment Score=1760

3.3 Corpus and evaluation metrics

Alignment is mostly utilised in parallel corpora to find the relation between sentences which have some similar content. In multilingual corpora relations can be that sentences are translations of each other in different languages. In monolingual corpora it can be of entailment. We have taken a monolingual corpus built for fusion which has 300 sentence pairs [128]. These sentence pairs are fetched from news articles and each pair contains information about similar events. This corpus is primarily build for generating new sentences by fusing these sentence pairs together. We choose this corpus as it has long sentences which make it suitable for testing combinatorial explosion issues in the original alignment algorithm when the number of words increases in the sentence. We align the sentence pairs using both the old alignment method and the new proposed approach with clause splitting. The results are compared in terms of time taken to perform alignment and the final alignment score generated from both of these methods.

The time taken to align sentences is stored for every pair to align and results from the original algorithm are taken as baseline. Results from the proposed methodology using clause splitting are compared against the baseline results. Time ratio is computed from the following formula.

$$\text{Time ratio} = \frac{\text{Alignment time}_{original}}{\text{Alignment time}_{clause}} \tag{3.3}$$

Time ratio >1 signifies improvement in efficiency.

To check the effect of clause splitting on alignment, the alignment score from the baseline approach is compared with the final alignment score from the new method. The alignment ratio is computed from the following formula.

$$\text{Alignment ratio} = \frac{\text{Alignment Score } clause}{\text{Alignment Score } original} \tag{3.4}$$

If alignment ratio>1 then it signifies improvement in alignment score, If alignment ratio <1 then it signifies alignment loss.

3.4 Results and analysis

Experiments were carried out on the above described fusion corpus of 300 sentence pairs. Alignment time and alignment score from the original algorithm are stored as results with the original alignment. The same metrics with the new approach of clause alignment are stored as clause alignment. Implementations of algorithms are done in the Java platform with CoreNLP APIs from Stanford. The machine used for this experiment has Intel core i5 processor with 6 GB RAM. In Table 3.2 we have presented sentence pair from the corpus, the generated clauses after clause splitting, their alignment and time performance. We analyze it manually in Table 3.2 on a few sample pairs and then present the overall ratio after running complete experiments. In the alignment list, words along with their position in the sentence are shown as aligned nodes.

The results of sentence pair 1 are presented in the column (Time, Score) of Table 3.2 and show that it takes a shorter time after clause splitting than with original algorithm. In this case alignment accuracy is 100% as measured by *Alignment Ratio*=2134/2120=1.01. This sentence pair is an example of preventing alignment loss by considering the second best alignment while combining clause alignments. In this case the first sentence has a clause

Table 3.2: Alignment performance of few sample pairs.

	Sentence pair to align	Original alignment, Clause alignment	Time(milliseconds), Score
1	1:Madoff told the employees he was finished, that he had absolutely nothing, that it's all just one big lie and it was basically, a giant (pyramid) scheme, according to the complaint filed in court.	*Original alignment*= [that9--Madoff1, he10--Wednesday9, had11--told2, absolutely12--senior3, that15--that10, it16--it11, 's17--'s12, all18--all13, just19--just14, one20--one15, big21--big16, lie22--lie17, it24--it20, was25--was21, basically26--that19, a28--a24, giant29--Ponzi26, pyramid31--giant25, scheme33--scheme27, to36--to40, the37--the41, complaint38--complaint46, court41--criminal45]	2661, 2120
	2.Madoff told senior employees of his firm on Wednesday that it's all just one big lie and that it was basically, a giant Ponzi scheme, with estimated investor losses of about $50 billion, according to the U.S. Attorney's criminal complaint against him	*Clause Alignment*= [that9--Wednesday9, he10--his6, had11--told2, nothing13--Madoff1, that15--that10, it16--it11, 's17--'s12, all18--all13, just19--just14, one20--one15, big21--big16, lie22--lie17, it24--it19, was25--was20, basically26--that18, a28--a23, giant29--giant24, pyramid31--Ponzi25, scheme33--scheme26, to36--to39, the37--the40, complaint38--complaint45, court41--him47]	1093, 2134
2	1:Jackson showed no reaction to the testimony, which focused on allegations that he gave alcohol to children and conspired to hold the accuser's family captive to get them to rebut the TV documentary, in which the boy and his siblings appeared and in which he said he let children sleep in his bed while he slept on the floor.	*Original alignment*= [gave15--elicited4, children18--was3, conspired20--conspired10, to21--to11, hold22--hold12, the23--the13, accuser24--accuser14, family26--family16, captive27--captive17, get29--get19, them30--them20, to31--to21, rebut32--rebut22, the33--a23, TV34--200325, documentary35--documentary26, which38--which28, his42--his37, appeared44--said30, the61--The1, floor62--testimony2]	45300, 2160
	2:The testimony was elicited to support allegations the singer conspired to hold the accuser's family captive and get them to rebut a February 2003 documentary in which Jackson said he allowed boys to sleep in his bed.	*Clause alignment*= [the6--The1, testimony7--testimony2, that13--boys32, he14--he30, gave15--allowed31, conspired20--conspired10, to21--to11, hold22--hold12, the23--the13, accuser24--accuser14, family25--family15, captive26--captive16, get28--get18, them29--them19, to30--to20, rebut31--rebut21, the32--a22, TV33--February23, documentary34--documentary25, in36--in26, which37--which27, his41--Jackson28, appeared43--said29, his54--his36, bed55--bed37]	6340, 2349
3	1:The White House sought to play down Roberts' participation in the case, known as Romer vs. Evans, in which the Supreme Court voted 6-3 in 1996 to strike down a voter-approved Colorado	*Original alignment*= [The1--the3, sought4--oversight9, the12--the14, case13--litigator16, Romer17--Romer18, Evans19--Evans20, to30--which22, strike31--struck23, down32--down24, a33--a25, voter-approved34--voter-approved26, Colorado35--Colorado28, initiative36--initiative29, that37--that30, would38--would31, have39--have32, allowed40--allowed33,	10489, 2880

	initiative that would have allowed employers and landlords to exclude gays from jobs and housing.	employers41--employers34, landlords43--landlords36, to44--to37, exclude45--exclude38, gays46--gays39, jobs48--jobs41, housing50--housing43	2910, 2760
	2:Smith said the omission was probably just an oversight because Roberts was not the chief litigator in Romer vs. Evans, which struck down a voter-approved 1992 Colorado initiative that would have allowed employers and landlords to exclude gays from jobs and housing.	*Clause alignment*= [Roberts8--Roberts11, participation9--litigator16, the11--the14, Romer16--Romer18, Evans18--Evans20, to29--which22, strike30--struck23, down31--down24, a32--a25, voter-approved33--voter-approved26, Colorado34--Colorado28, initiative35--initiative29, that36--that30, would37--would31, have38--have32, allowed39--allowed33, employers40--employers34, landlords42--landlords36, to43--to37, exclude44--exclude38, gays45--gays39, jobs47--jobs41, housing49--housing43]	
4	1:Yesterday, Daschle withdrew his name after acknowledging he paid $146,000 in back taxes and interest	*Original alignment* = [Yesterday1--the5, withdrew4--came3, his5--His1, name6--undoing2, $11--paid19, 146,00012--146,00021, back14--back23, taxes15--taxes24, interest17--interest26]	360, 720
	2:His undoing came with the release of his financial disclosure forms last Friday and information that he had paid $146,000 in back taxes and interest to resolve problems flagged by Obama's vetters	*Clause alignment*= [his5--His1, name6--undoing2, $11--paid19, 146,00012--146,00021, back14--back23, taxes15--taxes24, interest17--interest26]	649, 720
5	1:Asked about his beliefs during his 2003 Senate confirmation hearing, he replied that Roe was the settled law of the land.	*Orignal alignment*= [his6--his6, 20037--20037, Senate8--his11, confirmation9--confirmation8, hearing10--hearings9, he12--he2, replied13--told3, that14--that16, Roe15--the17, was16--was19, the17--the20, settled18--settled21, law19--law22, the21--the24, land22--land25]	390, 1520
	2:But he told senators during his 2003 confirmation hearings for his current appellate court post that the decision was the settled law of the land.	*Clause Alignment*= [Asked1--told3, his3--his11, his6--his6,Senate8--20037, Confirmation9--confirmation8, hearing10--hearings9, he12--he2, that14--that16, Roe15--the17, was16--was19, the17--the20, settled18--settled21, law19--law22, the21--the24, land22--land25]	698, 1560

(1.1: that it's all just one big lie and it was basically, a giant (pyramid) scheme) which maps to two clauses of second sentence *(2.1: that it's all just one big lie and 2.2: that it was basically, a giant Ponzi scheme, with estimated investor losses of about $ 50 billion).* Due to the new combination algorithm both alignments from both clauses(2.2 and 2.3) are present in the final alignment *[…it16--it11,'s17--'s12, all18--all13, just19--just14, one20--one15,big21--*

big16,lie22--lie17,it24--it19,was25--was20,basically26--that18,a28--a23,giant29--giant24,pyramid31--Ponzi25,scheme33--scheme26,….] and there is no alignment loss.

In pair 2 and 3 we see that the original time for alignment increases exponentially to 45300 milliseconds and 10489 milliseconds respectively. After clause alignment the time for pair 2 and 3 is reduced to 6340 milliseconds, 2910 milliseconds respectively. This is a significant reduction in time taken to align such sentence pairs. Results of pairs 4 and 5 indicate that clause splitting increases alignment time for short sentences. This confirms that clause splitting benefits only long sentence pairs containing more than 40 words. Alignment ratios of all pairs are more than 1, which implies no alignment loss during clause splitting.

3.4.1 Overall time ratio

Table 3.3 shows the alignment results. Overall time ratio computed from Equation (3.3) for the complete corpus is 1.13. This shows the improvement in time performance for sentence alignment from the new proposed method. We manually analysed the sentence pairs for which time performance improved from clause alignment method and can conclude that 20% of sentence pairs from this corpus benefitted from clause alignment in time performance. These pairs contain around 40 word long sentences. The remaining 80% pairs for which we did not experience any improvement in time are less than 40 words long. This analysis indicates that this approach is useful for only lengthy sentences.

In sentence pairs, which have fewer clauses and were originally taking much less time due to fewer nodes in the dependency parse tree, an increase of time in observed after clause alignment. This increase could be due to operations of all clause mapping and combining alignment. However increment in these cases does not affect the results much, as the original time and clause time is less than 1 second in these cases.

Table 3.3 Alignment results

	Time Ratio	Alignment Ratio
Mean	1.13	1.07
Std. Deviation	2.28	1.09

3.4.2 Overall alignment ratio

The overall alignment ratio computed by Equation (3.4) is 1.07. This ratio indicates improvement in overall alignment of sentence pairs. Analysis of the alignments shows that 20.9% pairs had less alignment scores than achieved by original algorithm. The mean loss observed in these pairs is 11.28%.

Manual analysis of the pairs in which alignment loss occurs shows that sometimes a combined clause alignment does not turn up the way the original sentence to sentence alignment works. There are a few cases when words do not have same dependency relation to each other in clauses as they had in original sentence. This is due to dependency parsing errors.

Another reason is inefficient clause splitting. In this case a clause gets highly aligned to two or more clauses of other sentence. This happens either because first the clause did not break correctly into further clauses or it is very similar to many clauses. This makes the combined alignment erroneous. It can be improved by incorporating more POS tags into clause splitting process or by utilizing better available approaches for this task.

However for the majority of sentence pairs we experience no loss and some improvement in the alignment.

3.5 Conclusion

In this Chapter we have described our work on sentence alignment and we have evaluated the performance of an existing popular approach which utilizes dependency tree representation of sentences. We have proposed that by dividing sentences into smaller fragments- *clauses* – we can make performance of the older method faster. As the same time we have proposed ways to preserve the original intended alignment. Experimental evaluation favors the proposed method as the time reduces for the sentence more than 40 words long. We have shown the implementation of our proposed method by utilising existing NLP resources, which makes the proposed methodology viable. Further extension to this work should include improved ways of clause splitting by utilizing more semantic and syntactic information. For node similarities further concept based similarity measures can be utilized to make the alignment more accurate.

Pseudocode

```
Split (String sentence)
{
parseTree=Parse(Sentence)  //we have used Stanford Syntactic Parser
RootNode=get_ root_node (parseTree);
create empty lists ListOfClauses, ListOfWords;

Initialize a global variable SBARFlag=0   /* this flag is increased if the POS tag of current rootNode is
                                SBAR Clause */

getClauseStrings(RootNode, ListOfWords , ListOfClauses);  //call to clause generating  function
return ListOfClauses;
//A preprocessing may be required to remove single word clauses // (i.e. and, '.')
}

getClauseStrings (TreeNode, ListOfWords, ListOfClauses){
/*this function generates the clauses according to the POS tags for clause level*/
List of children ChildList=TreeNode.getChildren();
For all children in ChildList
ChildNode=TreeNode->nextChild();
If childNode is a Leaf Node
                        get the label of TreeNode;
                        add the Label in ListOfWords;
```

```
else If POS_tag(childNode) ==”S” and (SBARFlag <=0)  /*check if we are not  already
                                                          considering a SBAR clause*/

getClauseStrings(childNode, ListOfWords, ListOfClauses );
//recursive call to itself
For all words in ListOfWords
Add words to clause string C with space;
                    //generate a clause by adding all words
Add C to ListOfClauses;
Clear ListOfWords;
else If POS_tag(childNode) == “SBAR” or “SBARQ”
SBARFlag= SBARFlag+1; //entered into a SBAR clause
Create a new temp_ListOfWords;
/*a new list of words is created because we consider words only unde SBAR tag to be included in this clause not the words before SBAR*/

getClauseStrings(childNode, temp_ListOfWords, ListOfClauses );
//recursive call to itself
For all words in temp_ListOfWords
Add words  to clause C with a space
                    Add C to ListOfClauses;
                    Clear temp_ListOfWords;
else
getClauseStrings(ChildNode,ListOfWords, ListOfClauses );
//recursive call to itself
}
```

Chapter 4

Semantic Graphs and Text Summarisation

In the previous Chapter, we have presented our improved approach for sentence alignment, which is a useful technique for tackling redundancy in text summarisation. We moved further in the direction of summarisation by reflecting upon the research questions we have set for our work in order to contribute to text summarisation. Our research objective described in Chapter 1 is semantic representation of textual information and summary generation from this representation. The literature review of text summarisation in Chapter 2 shows that the best performing summarisers utilise graphical representation of text information among which the popular ones are LexRank[89], TextRank[90], SemanticRank[144] and UnifiedRank[95]. Analysis of past work in summarisation and related areas of information retrieval shows that graphical display can open up ways for better analysis of information by incorporating graph based popular ranking methods such as *PageRank*, *HITS* into the summarisation process. Graph/Network analysis is already popular in social network analysis. In this Chapter, we describe our research work to generate graphical representation of text from dependency relations between words in the sentences. We have analysed a popular methodology and proposed enhancement by incorporating more semantic information.

4.1 Graph based text summarisation

Various representations of textual information have been subject to analysis in automatic text summarisation. The main purpose of constructing these representations is to identify the

important concepts hidden in the document. Importance scores of concepts pave the way for extracting important sentences to generate the summary. Graph based summarisation methods are categorised into either lexical or semantical approaches. Lexical approaches view the text document as a set of words and analyse it based on words' positional proximity i.e. n-grams or relations derived from a common sequence of words. In these approaches mostly *stopwords* are removed and lexical relations are derived from only the words which may fall into particular syntactical categories i.e. nouns. In the summarisation method of LexRank [89] intra sentence similarity is considered a lexical relation between two sentences. In this approach, sentences are converted to vectors of words and then cosine similarity is calculated between these vectors. By considering each sentence as a node of the graph, these similarity scores connect the nodes. Two graph based centrality measures, degree centrality and eigenvalue centrality are applied to the stochastic matrix derived from adjacency matrix of this sentence similarity graph, to extract the most salient sentences as the summary. In another approach directed syntactic representation of words is used for supervised and unsupervised keyword extraction [94]. In this kind of a graph words are connected based on their co-occurrence in the document. Extracted keywords are later used for extracting summary sentences. Similar kinds of approaches were followed in TextRank[90] where vertices of graphs are unique sentences from the document and relations between vertices is the degree of content overlap between connected sentences in the document. Opiniosis summariser [182] uses word graphs made of adjacency relations between words for opinion summarisation. Edge weights in these word graphs are the frequency of co-occurrence of those words in the document. A recent word graph based approach which is based on a bi-gram co-occurrence relation taken from the document and from Wikipedia extended abstracts have used weighted minimum vertex scores of words in the word graphs to calculate sentence importance score [93]. All these graphs are based on syntactic, lexical and positional

properties of words in sentences. However, the meaning of sentences cannot only be interpreted in terms of order of words. It requires deeper relations to be identified between long distant words. A lexical graph fails to exploit the relations between long distant words and thus needs deeper semantic relations to be added to the graph.

Relations between words or phrases which are associated to meaning of words or conceptual similarity between them and which are generally beyond lexical comparisons are called semantic relations. Semantic relations derived from meaning of words either can be a synonymy relation between two non-similar lexical words or can be hyponymy relation when two different words share a common root. Another kind of semantic relation is the dependency relation, which is derived from inter-dependency between roles of words in sentences to express the desired meaning. We have described dependency parsing in Section 3.2.1 of Chapter 3 on sentence alignment. Most common dependency relations are of *subject*, and *object* of a transitive verb. Graphs of textual information of documents, which integrate these semantic relations, are termed as semantic graphs.

Semantic graphs were part of previous summarisation research. The WordNet ontology has been extensively utilised to check for possible semantic relations between words and to form a semantic graph based on these relations. In these approaches, sentences are divided into terms to map them to *synsets* of its stem word in WordNet ontology. Then hierarchical structure from the ontology is used to connect the words of the document[22] and further analysis is performed considering the document as a subtree in this ontological hierarchy. Tree based similarity measure are applied to this representation to identify summary sentences using supervised or unsupervised approaches [102]. The *Semantic Rank* summariser combines WordNet ontology and Wikipedia knowledge base to approximate similarity between sentences[144]. Scores from this combined similarity measure are taken as weights of edges connecting the sentences as vertices.

The most prevalent approach to construct semantic graphs for text summarisation has been by exploiting logical dependency relations of words. Dependency relations provide insight about the roles of words with respect to the predicate (verbs) or to each other. Generally, logical triples of *subject-predicate-object* (also referred as SVO triples) are extracted from documents using syntactic and dependency parsers and connected as a graph. Later triples are merged based on ontological relations as explained previously to form better connected graphs [67,68,183]. Dominance of these kind of semantic graphs in text summarisation can be attributed to the fact that *predicate* signifies the main action discussed in the sentence and its main arguments are actor (*Subject*) and receiver of action (*Object*). Hence logical triples are expected to represent the important information content and thus it forms the basic units of semantic graph representation in popular text summarisation methods[184,149]. In the research study presented in this Chapter we have developed semantic graphs based on logical triples from dependency parse of sentences and analysed the information links and its impact on summarisation. From our analysis we have proposed a novel semantic graph generation method which uses more semantic information from dependency relations than only *Subject-predicate-object* triples and have analysed the performance of new semantic graphs on text summarisation. The key idea in the improved approach is to look for connected words through long distance dependency relations which can be made of sequence of dependency relations. A similar idea has been used in information retrieval for identifying relations between terms[108], but has not been utilised in summarisation previously.

4.2 Semantic graph construction from logical triples

The first application of automatic extraction of triples in text summarisation has been analysed by Lescovec et al. [149]. They have used a propriety tool NLPWin from Microsoft

to directly get triples for sentences and train classifiers to learn summary subgraphs. Extraction of triples was later researched using open-source syntactic parsers. Few open-source parsers utilised for triple extraction were Minpar parser, Stanford syntax parser and openNLG parser. We have implemented extraction of triples for our analysis using the Stanford dependency parser. We have also incorporated deeper linguistic analysis i.e. co-reference resolution and named entity recognition as done in previous approaches on semantic graphs. Deeper inguistic analysis works on the knowledge of language , dictionary and finds out the strucuture of the text. It can find out the deep facts such as two references to same entity in the sentences(coreference resouluton) which only machine learning approaches that works on the bag of words approaches may miss.

A sample logical SVO triple extracted from a simple English sentence *the cat sits on the mat* is *cat->sit->mat.* A semantic graph from triples of two sentences *"Sam brought the cat. The cat sits on the mat." i*s shown in Figure 4.1.

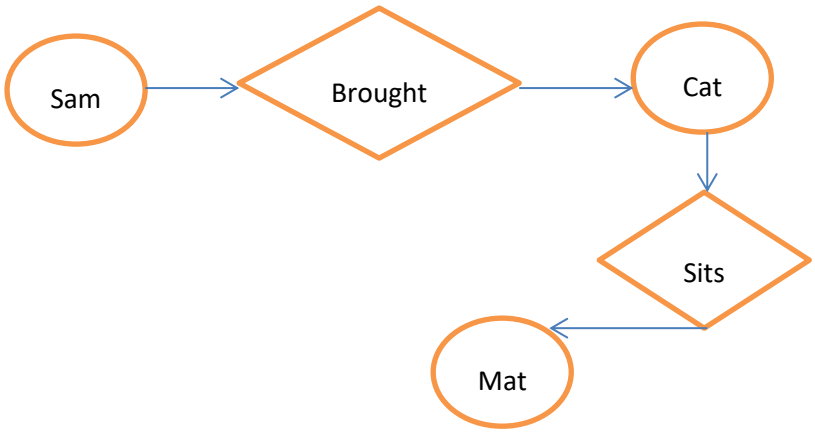

Figure 4.1: A semantic graph of two sentences.

To extract the triples from sentences we identify patterns of dependency relations from the dependency parse tree of that sentence. Following patterns are utilised in this process.

1. nsubj-verb-dobj

This dependency relation pattern matches the dependencies in the sentence *Sam bought a cat* and yields triple *Sam-bought-cat.*

2. nsubj-verb-iobj

This dependency relation pattern matches the dependencies in the sentence *Sam gave a book to Lisa.* and yields triple *Sam-Gave –Lisa.*

3. nsubj-verb-prep_obj

This dependency relation pattern matches the dependencies in the sentence *Sam ordered food from Chinese restaurant* and yields the triple *Sam-ordered_from–restaurant* along with triple *Same- ordered- food.*

The following patterns are applicable to passive voiced sentences

4. agent-verb-nsubjpass

5. agent-verb-prep_obj

After extracting the triples from the text by utilising the dependency relation patterns the nodes in the triples are merged based on lexical similarity and co-reference information. In next Section we describe the co-reference resolution method to resolve the references in the original text and then we analyse the triple-based semantic graph in detail.

4.2.1 Co-reference resolution

The semantic graph of a document is expected to contain enough information from the original document to be used as its substitute in a text summarisation process. In abstract summarisation it can be used for generating sentences without the need for the original document. In extractive summarisation a semantic graph is mostly used for ranking entities, so salience of sentences can be estimated from the ranked entities. Thus it is important to

preserve the links between different sentences to make a connected graph. It requires linking the main entities of the document irrespective of the lexical word used to refer to them throughout the document. This can be pronominal reference or different nominal references. We have used Stanford's anaphora resolution software to generate the list of mentions to the same entities. It is not a straightforward task to replace the mentions with the head reference in all occurrences. Below an example of list of references is given.

In implementation and development of logical triple based semantic graph we have resolved references when entity is of type *location*, *person* or *organisation*. Below we present some examples and then the pseudocode of our algorithm for replacing the references in Figure 4.2(i) and Figure 4.2(ii).

A mention can start from bigger sequence of words where it contains adjectives, appositions to the minimum sequence of headwords. We avoid replacing these kinds of recurring references by checking the word span of reference and sentence number. We perform named entity recognition and create a map of word number to named entity for each sentence. Whenever the headword of the main reference has a mapping to a named entity, we replace further references with the named entities. In the following example text we describe the case of multiple mentions of the same referenced entity and the modified text after resolving co-references.

Original Text: *John Smith, a young police officer saw a van and went up to it. He grabbed the handle, but the van started to move. Mr. Smith will look into the issue.*

Mention_list1= ["John Smith , a young police officer" in sentence 1, "John Smith" in sentence 1, "a young police officer" in sentence 1, "He" in sentence 2, "Mr. Smith" in sentence 3], Representative mention= John Smith, pos= NNP, Rep headword=Smith11, Rep NE tag= PERSON

Mention_list 2= ["a van" in sentence 1, "it" in sentence 1, "the van" in sentence 2], Representative mention= van, pos= NN, Rep headword=van91, Rep NE tag= O

Mention_list 3 = ["the" in sentence 2], Representative mention= the, pos= DT, Rep headword=the22, Rep NE tag= O

Mention_list 4 = ["the issue" in sentence 3], Representative mention= issue, pos= NN, Rep headword=issue63, Rep NE tag= O

After resolving co-references from the algorithm described in pseudocode format in Figure 4.2(i) and Figure 4.2(ii) we get the following text.

John Smith, a young police officer saw a van and went up to van. John Smith grabbed the handle, but the van started to move. John Smith will look into the issue.

mention_list contains all mentions to the same reference across the different sentences of a document

Every *mention_list* has a representative mention ***rep_mention*** with ***head_index*** indicating the headword of that mention. ***rep_mention*** is considered the main referenced entity.

For each *mention_list*

    Find the lexical name of entity in *rep_mention* to be used for replacing other pronominal references from the same mention list.

        If *head_index* maps to a recognised named entity from the Named_entity_list for that sentencenumber

            Lexical_name=Named_entity_list[head_index]

        Else for(i=head_index, i>=0;i--)

            If(word[i] has same POS tag and Entity type

                Lexical name=word[i] + " "+lexical name//combining compound nouns

Figure 4.2(i): Pseudocode for anaphora replacement in sentences.

For each mention ***current_mention*** in ***mention_list***

If current_mention is part of rep_mention

Do not replace;

move to next mention;

Else If dependencyRelation (current_mention , rep_mention)=subject || dependencyRelation (current_mention , rep_mention)=apposition

Do not replace reference,

Move to next mention,

*// this is for cases like **He is John, my friend**. Although **he, my friend** both are references to **John**, nothing should be replaced here.*

Else If current mention is pronoun

Replace with lexical_name//make sure to check add 's in case of possessive pronouns

Else If headword of proper name is same as rep_mention head word and the reference is shorter from rep_metion replace with lexical name/// this if for short abbreviations of names

Replace proper nouns

Figure 4.2(ii): Pseudocode for anaphora replacement in sentences.

4.2.2 Analysis of triple based semantic graphs

Figure 4.4 shows the triple based semantic graph generated from our implementation for sample text shown in Figure 4.3. Analysis of the implemented triple based semantic graph shows that entities type of location or time are often linked through pre or post modifying

prepositional connectors to other entities. Thus a triple based graph which focuses on subject-predicate-object relation loses these entities and the links. Due to this reason we see the sparsity in the graph shown in Figure 4.4. In past summarisation approaches to reduce the impact of information loss in graphical representation of textual information the descriptive information about semantic graph nodes were added as features of classifiers. We hypothesize that bringing out this substantial information as semantic nodes, which may not be covered in basic triple based graph, will influence the ranking of semantic nodes. Consequently, it will affect summary extraction method by improving the scoring of sentences. In our research we have analysed this hypothesis by incorporating more dependency relations into semantic graph. Detailed analysis of triple based graphs exposes the loss of information, which is explained in Section 4.2.2.1, Section 4.2.2.2 and Section 4.2.2.3.

The resort town's 4,700 permanent residents live in Long Valley, a 19-mile-long, 9-mile-wide volcanic crater known as a caldera. Eruptions somewhat smaller than Mount St. Helens' happened 550 years ago at the Inyo craters, which span Long Valley's north rim, and 650 years ago at the Mono craters, several miles north of the caldera.

Figure 4.3: Sample text.

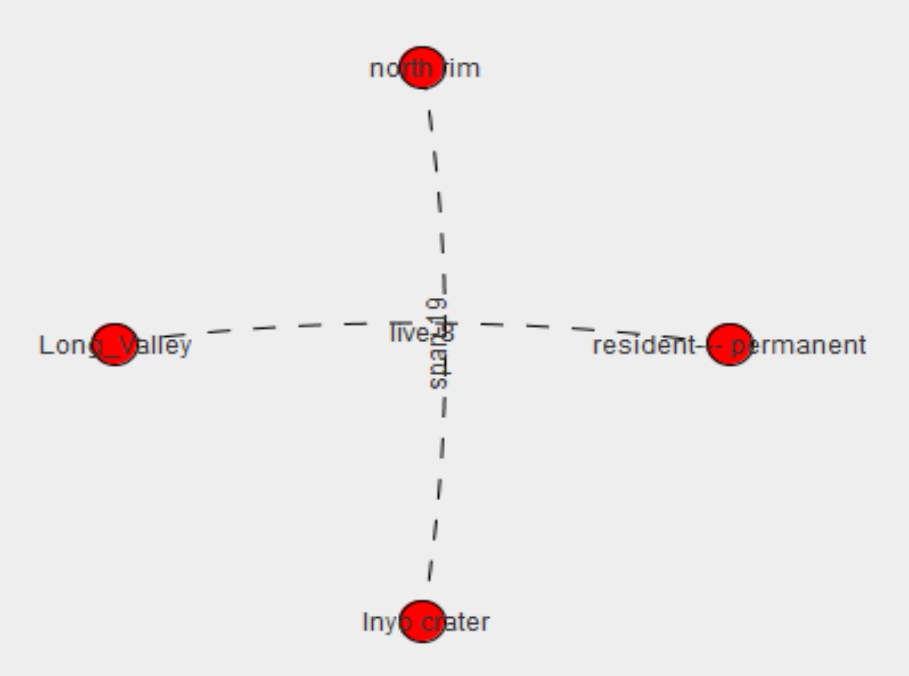

Figure 4.4: Triple based semantic graph of sample text shown in Figure 4.3.

4.2.2.1 Loss of links between words in sentence

Often links between named entities do not become a relation in a semantic graph because these entities were never subject/object for any predicate. In previous graph generation approaches, this information were combined with the feature sets which are derived from semantic graphs. This loss of information is described from the following text example.

Sentence*: President Obama's arrival in London created a joyful atmosphere.*

Triple generated for this sentence: *Arrival->create->atmosphere*

Here entities *London* and *Obama* are added to the feature set of semantic graph node *Arrival* and other information *Joyful* is added as feature to semantic graph node *Atmosphere*. We can observe here that important entities such as London did not become a node in the semantic graph because it was not part of a SVO triple. Also due to this a link is missing in the semantic graph between entity *London* and semantic graph node *atmosphere*. However, we

can manually observe that it is the atmosphere in London, which is talked about in the sentence. But this fact is not represented through triple based semantic graph. If we construct a graph using a sequence of dependency relations instead of direct relations then we can incorporate the missing link through the following sequence.

*London-**prep-in**->Arrival-**nsubj**->created-**dobj**->atmosphere*

4.2.2.2 Loss of inter-sentence links between words

Coherence between sentences is preserved by referring to common entities. In some sentences these entities are not covered in the subject/Object type and thus the other information present in these sentences becomes isolated at the document level representation of semantic graph. We can see this in the following example sentences from a text document:

He went to church in Long valley.

One of the explosions happened in Long Valley.

The triple generated for these sentences are:

He->went>church

Explosion->happened->long valley

A semantic graph constructed from both sentences will join these triples but will result in a sparse graph because the only shared entity *Long Valley* is not present or linked to first triple.

4.2.2.3 Identification of subject is not clear

Less precision is observed in parsing categories such as subjects of predicates when dealing with long distance dependency in complement clauses attached to verb phrase or to adjectival phrase[185]. This dependency is named "*xcomp*" and in this case a subject is not present in

the clause but determined from an external subject. Inaccuracy in determining subjects to clausal predicates leads to loss of connections in the semantic graph.

This analysis indicates that relying on the SVO triple based semantic graph for NLP processing may not be accurate as it is not a complete representation of the information to be ranked for further tasks. We proposed a solution to cover more information in the form of *dense semantic graph* which is described in the next Section.

4.3 Dense semantic graph

We have shown in the above analysis of the triple based semantic graph that due to loss of connections between information we end up with a sparse graph representation of a text document. Although we have other information at our disposal, we cannot take it directly from the graph. That leaves the representation inefficient and futile. So we decided to construct a dense graph that covers more information from the text document. We have seen that instead of a direct dependency relation if we consider a sequence of dependency relations then it connects the entities and preserves the coherent structure of the original document in the semantic graph. To preserve the coherency we have extended original triple based graph by incorporating more dependency relations into it. We extend it by developing a dense semantic graph from shortest distance dependency relations between words in the sentence after resolving co-references.

We formally describe a dense semantic graph G= (V, E), where

$$V = \left\{ \bigcup_{word_i \in document} Word_i : \text{pos}(Word_i) \in \{\text{JJ}*, \text{NN}*\} \right\} \quad (4.1)$$

In (1) $\text{pos}(Word_i)$ provides part of the speech tag of$Word_i$. According to the Penn tag set "JJ" signifies Adjectives and "NN" signifies Noun.

$$\text{Edge set } E = \{\bigcup_{u,v \in V}(u,v) : SD(u,v) \leq limit\} \quad (4.2)$$

In Equation (4.2) $SD(u,v)$ is the shortest distance from u to v in the dependency tree of that sentence and limit is the maximum allowed shortest path distance, which is varied from 2-5 in our experiments.

We implement this by first generating a temporary dependency graph for the document. We have used the Stanford dependency parser for generating dependency relations and *Java Universal Network/Graph Framework* (JUNG) for generating temporary tree graph. JUNG is graph visualisation software developed in the JAVA platform and provides easy access to graph visualisation and network analysis APIs. Words are converted to root form and only one node is added to the semantic graph for each unique lexical word. This ensures connectivity in the graph. We apply Dijkstra's shortest path algorithm on this tree graph to find the shortest path distance between all nodes. The final *dense semantic graph* is constructed by finding the vertices from this tree graph by applying Equation (4.1) and edges are determined by applying Equation (4.2). In detail we take nodes which are distant by some limited distance based on a shortest distance calculation. We restrict the words to be of syntactic type noun or adjective to become node in the dense semantic graph. The prime reason for this restriction is the application of semantic graphs in the text summarisation task and in the summarisation literature it has been observed that sentences are ranked from the scores of the noun nodes words in it rather than predicate words. Nouns are considered significant units of information compared to predicates and predicate nodes are never used for sentence scoring. The reason for including adjectives is that they present modification information of nouns and play a significant role in distinguishing one instance of a noun from another similar noun.

Before construction of dense semantic graph we pre-process the text document to replace the pronominal and nominal references with the references of main entities. The same algorithm

which we have developed for anaphora replacement in triple based semantic graph, explained in Section 4.2.1 has been followed here. We have employed the natural language processing toolkit Stanford's CoreNLP for pre-processing and semantic graph construction. The pipeline followed is part of speech tagging, stemming, named entity recognition, anaphora resolution and dependency parsing[186,187,30]. The final dense semantic graph is developed in the JUNG software [15] to analyse its application in the text summarisation task. Salience of semantic nodes is determined by applying a graph ranking algorithm for selecting summary sentences from this ranking.

For the text fragment shown in Figure 4.3, a dense semantic graph has been generated from our implementation and is shown in Figure 4.5.

The key difference between two types of graphs is that *Triple based semantic graph* consists of only 3 direct dependency relation between words-*subject, verb* and *object*, whereas *Dense semantic graph* considers many indirect dependency relations by varying the distance paramenter. Comparing both triple based semantic graphs in Figure 4.4 with the dense semantic graph in Figure 4.5 we see the difference of coverage of information in both graphs. Triple based graph has four nodes, which covers four entities, whereas the dense graph has 19 nodes among which six entities and four more nouns are covered. This shows the high coverage of dense semantic graphs. In next Section we evaluate the performance of both semantic graphs in the text summarisation task.

4.4 Summary generation from semantic graphs

Our aim of research is to evaluate the efficiency of semantic graphs in text summarisation as an intermediate representation. Two semantic graphs were developed in this research work and we have experimented on them. We compare the summary qualities generated from both

of the approaches which differ in only the representation of text. Summary generation starts by first approximating the salience of nodes in the semantic graphs. For this we have used one of the popular graph ranking algorithms, *PageRank.*

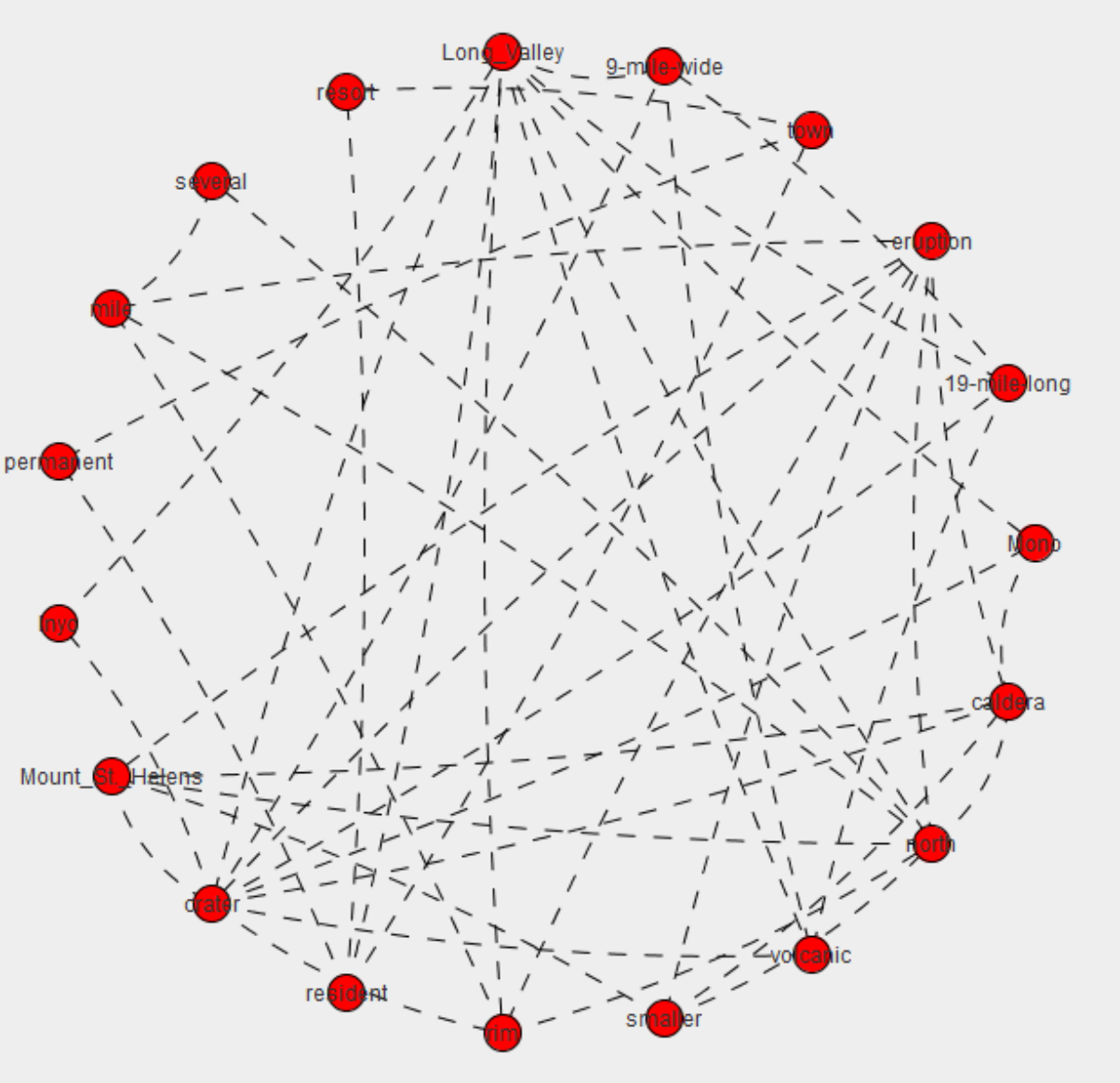

Figure 4.5: A dense semantic graph.

4.4.1 PageRank

PageRank is a popular graph analysis method used by the google search engine to give authority scores to web pages in the web information retrieval[188,189]. To calculate the score

of a web page this method takes into account the authority scores of all pages from which there are incoming links towards this page and the count of all outgoing links from these pages.

This is different from degree centrality where the count of incoming links determines the importance of a node in the network because *PageRank* not only considers incoming links but also considers the importance of nodes which these incoming links are coming from. The *PageRank* score of nodes in a network are calculated by analysing the non-negative transition matrix of the network. The transition matrix is the probability of going from one node to another node. For a simple directed graph with adjacency matrix A, its transition probability matrix B can be represented as

$$B_{i,j} = \frac{A_{i,j}}{\sum_k A_{i,k}} \tag{4.3}$$

PageRank score PR_{nodei} of a node $node_i$ as the sum of equally distributed *PageRank* scores of all incoming nodes.

$$PR_{node_i} = \sum_{node_j \in In(node_i)} \frac{PR(node_j)}{deg(node_j)} \tag{4.4}$$

It can be written in the matrix form by $\mathrm{Pr} = \mathrm{B}^{\mathrm{T}}\mathrm{Pr}$

To accommodate nodes which do not have outgoing links a random jump is required thus a damping factor d is introduced in the *PageRank* calculation.

$$PR_{node_i} = (1-d) + d * \sum_{node_j \in In(node_i)} \frac{PR(node_j)}{deg(node_j)} \tag{4.5}$$

The dampening factor d is set between 0 and 1. Generally its value is set to 0.85. A random walker on this Markov chain chooses one of the adjacent states of the current state with probability d, or jumps to any state in the graph, including the current state, with probability (1-d).

Equation (4.5) can be rewritten as in Equation (4.6).

$$\mathrm{Pr} = ((1 - \mathrm{d}) + \mathrm{d} * \mathrm{B})^{\mathrm{T}}\mathrm{Pr} \tag{4.6}$$

Suppose the new transition probability matrix after incorporating a dampening factor is Q.

$$\mathrm{Q}=(1 - \mathrm{d}) + \mathrm{d} * \mathrm{B}) \tag{4.7}$$

Then Equation (4.6) can be written as Equation (4.8).

$$\mathrm{Pr}=\mathrm{Q}^{\mathrm{T}}\mathrm{Pr} \tag{4.8}$$

Equation (4.8) is same as Equation (4.9).

$$\mathrm{Pr}^{\mathrm{T}}\mathrm{Q}=\mathrm{Pr} \tag{4.9}$$

Since Q is a non-negative square matrix *Pr* represents the dominant eigenvector of Q with corresponding eigenvalue 1. It is calculated by power iteration method applied on Q. Some arbitrary values are assigned to Pr initially which does not impact on the final convergence values of Pr.

4.4.2 Computing sentence scores from semantic graphs

In our text summarisation approach we apply *PageRank* method to triple based semantic graphs and dense semantic graphs. After applying it we end up with the *PageRank* scores of each node which indicates its importance based on its connections to other nodes in the graph. We calculate a word vector W_i for each sentence S_i corresponding to nodes of semantic graph, where:.

$$\mathrm{W_i}=(\mathrm{w_{io}}.\mathrm{w_{i1}}\ldots\mathrm{w_{ij}}) \tag{4.10}$$

j is the count of nodes in the semantic graph.

$$w_{ij}=\begin{cases}1\ if\ node_j\ is\ subsequenceOf(S_i)\ , \\ 0\ otherwise\end{cases} \quad (4.11)$$

From the scores of nodes and the word vectors we calculate the importance score $SentenceScore_{S_i}$ of each sentence S_i in the document as:

$$SentenceScore_{S_i} = \frac{\sum_{j=0}^{count(vertex)} PR_j\ .\ Wij}{length(S_i)}. \quad (4.12)$$

After computing the sentence scores, sentences are sorted in descending order of their score. A summary is generated from these sorted sentences depending on the permitted word length.

Since our test corpus is made of news corpora, and sentence position has been the most impacting factor for this genre of text information for summary selection we have decided to test the summarisation results using this feature. We included a feature "position of sentence" and compute new scores using following equation.

$$newSentenceScore_{S_i} = \frac{0.1\times(Count_{Sentences}-i)}{Count_{Sentences}} + 0.9 \times SentenceScore_{S_i} \quad (4.13)$$

In the experiments *on dense semantic graph*, we have varied the shortest dependency path length from 2 to 5. In this way, we have tested 4 variations of dense semantic graph with extractive summarisation.

4.5 Experiments

In this section we describe the corpus and different experimental setups for the analysis.

4.5.1 Corpus

We have used corpora from the DUC conference for our experimentation. DUC conferences has been organising single document summarisation tasks until the year 2002, after which they moved to guided single document summarisation and multi-document summarisation. This has made the DUC corpus the most experimented on one to date due to unavailability of any new standard corpus.

There are two corpuses DUC-2001 and DUC-2002 in the dataset. DUC-2001 contains 308 documents taken from news sites. Documents of DUC-2001 are divided in 60 topics. The other corpus DUC-2002 consists of 565 documents divided into 59 topics. Among these 28 documents are repeated twice in different topical categories, which make it 537 unique documents. Each document in both corpuses has two human written summaries by different authors, which are abstractive in nature. Summaries are approximately of 100 words length. Analyses of these human summaries shows that even humans have conflict of opinions in deciding which information should be included in a short summary for same document. Never in both the human written summaries could we find the same content. This has made summary evaluation difficult which is explained later in the evaluation Section after the setups are described in the following Section.

4.5.2 Setup

For each document we generate a triple based semantic graph and then apply *PageRank* analysis shown in Equation (4.5). After *PageRank* analysis of the semantic graph sentence scores are calculated using Equation (4.12) and Equation (4.13). We select summary sentences from sorted list of sentences until we reach 100 words. We remove any words after 100 words. The results of this setup are presented in *Triple based summarisation*.

Next we apply "*position*" feature to calculate new scores of sentences after *PageRank* analysis of the graph and select summary sentences by applying Equation (4.13). The results for this setup are presented in *triple + "position" summarisation*.

Similarly for experiments on dense semantic graph, we generate dense semantic graph for each document of DUC-2001 and DUC-2002 corpus. In dense semantic graph construction, we vary shortest distance between nodes from 2 to 5. We stopped after 5 because we did not see improvement in results after it. Results from pure *PageRank* analysis are presented as *Dense semantic graph summarisation*. Results are tagged with the distance from 2 to 5 and described as *Dense semantic Graph -2, Dense semantic Graph -3, Dense semantic Graph -4* and *Dense semantic Graph -5*. Results after including feature "position" feature with *PageRank* analysis on dense semantic graph are shown as *Dense semantic graph-distance+ Position*.

We have compared the performance of our system with publicly available summarisation system *open text summariser* (OTS). We have evaluated the results using n-gram comparison between referenced and system generated summaries using the ROUGE toolkit. The toolkit has been discussed in Section 2.5.1 of Chapter 2, which describes different evaluation measures of system generated summaries. Although there are conflicts in using the ROUGE measure as a standard evaluation system because it is purely a lexical matching of words or sequence of words which we cannot evaluate correctly if synonymous words have been used for summary generation. Nevertheless ROUGE scores has been considered close to human evaluation for extractive summarisation when we have more than one reference summaries as it increases accuracy of n-gram match. ROUGE scores have been generated on two setting *Stemmed words and stop words included* and *Stemmed words and stop words removed.*

4.6 Evaluation

Table 4.1 shows the ROUGE-1, ROUGE-2 and ROUGE-W scores for DUC-2001 data achieved by different experimental runs described in previous Section on ROUGE setting *Stemmed words and stop words included.* Table 4.2 describes results on the DUC-2001 corpus with Rouge setting *Stemmed words and stop words removed.* In Figure 4.6, Figure 4.7 and Figure 4.8 we have plotted the Rouge-1, ROUGE-2 and ROUGE-W score of Triple based summariser, *Dense semantic Graph-4 summariser, Dense semantic Graph-5 summariser* and *OTS summariser vs.* different rouge setting on DUC-2001 corpus. We observe that lowest Rouge scores are reported with triple based experiment. By including position in triple, scores are improved. In Table 4.1 and Table 4.2, we see that Rouge-1 scores for *Dense semantic Graph-2, Dense semantic Graph-3, Dense semantic Graph-4,* and *Dense semantic Graph-5* improve linearly and are better than *triple based summarisation* and *triple + position based summarisation*. This shows that as the shortest distance of dependency path was increased from 2 to 5, Rouge score has improved due to better ranking of the nodes in the semantic graph. This better ranking can be attributed to more connections found after increasing the path distance to find links in the dependency tree. Similar trend of increase in ROUGE-2 and ROUGE-W scores are observed for *Dense semantic Graph-2, Dense semantic Graph-3, Dense semantic Graph-4, Dense semantic Graph-5* and *Dense semantic Graph +position.* As seen in Figure 4.6, Figure 4.7 and Figure 4.8 although benchmark OTS results are always higher than the best results achieved by our Dense semantic graph approach, it can be observed that our results are comparable to their results.

Table 4.1: ROUGE-Scores on DUC-2001 corpus with ROUGE setting - stop words included.

Summarisation System	Rouge-1	Rouge-2	Rouge-W
Triple Semantic Graph	0.39106	0.13924	0.12410
Dense semantic Graph -2	0.40123	0.14470	0.12653
Dense semantic Graph -3	0.40178	0.14532	0.12710
Dense semantic Graph -4	0.40208	0.14617	0.12750
Dense semantic Graph -5	0.40008	0.14412	0.12680
Triplet Semantic graph +position	0.39653	0.14044	0.12551
Dense semantic Graph -4 + position	0.40682	0.15190	0.12964
Dense semantic Graph-5 + *position*	0.40568	0.15067	0.12940
OTS	0.42070	0.17236	0.12965

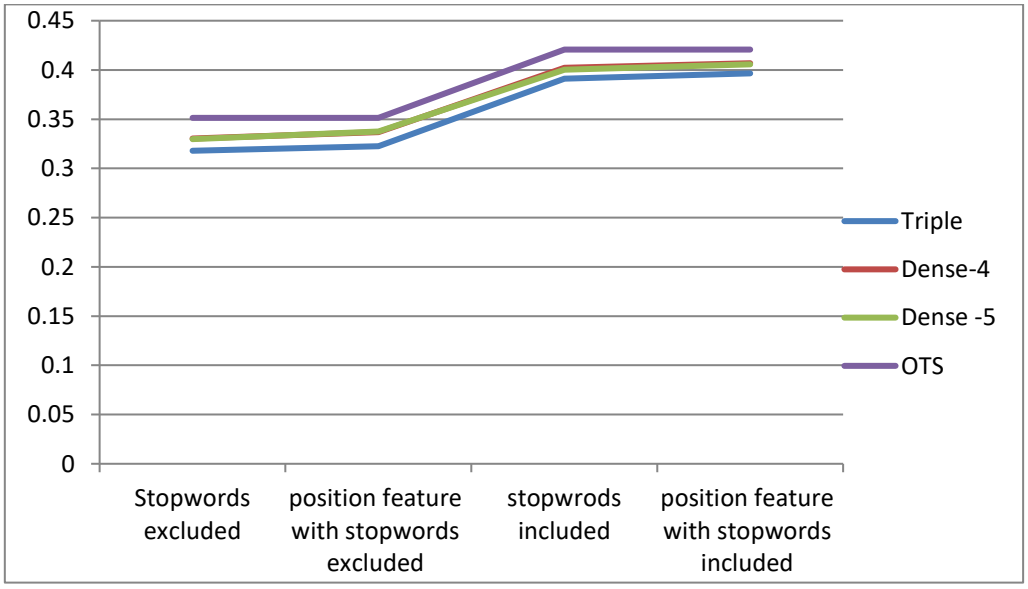

Figure 4.6: Rouge--1 score vs. different rouge settings and features on DUC-2001 corpus.

Table 4.2: ROUGE-Scores on DUC-2001 corpus with ROUGE setting - stop words removed

Summarisation System	Rouge-1	Rouge-2	Rouge-W
Triple Semantic Graph	0.31793	0.12829	0.13214
Dense semantic Graph -2	0.32964	0.13229	0.1354
Dense semantic Graph -3	0.3298	0.13301	0.1359
Dense semantic Graph -4	0.33037	0.1351	0.13671
Dense semantic Graph -5	0.32974	0.13365	0.13621
Triplet Semantic graph +position	0.3224	0.12923	0.13355
Dense semantic Graph -4 + position	0.33676	0.14106	0.14017
Dense semantic Graph-5 *+ position*	0.33753	0.14049	0.14023
OTS	0..35134	0.16039	0.14093

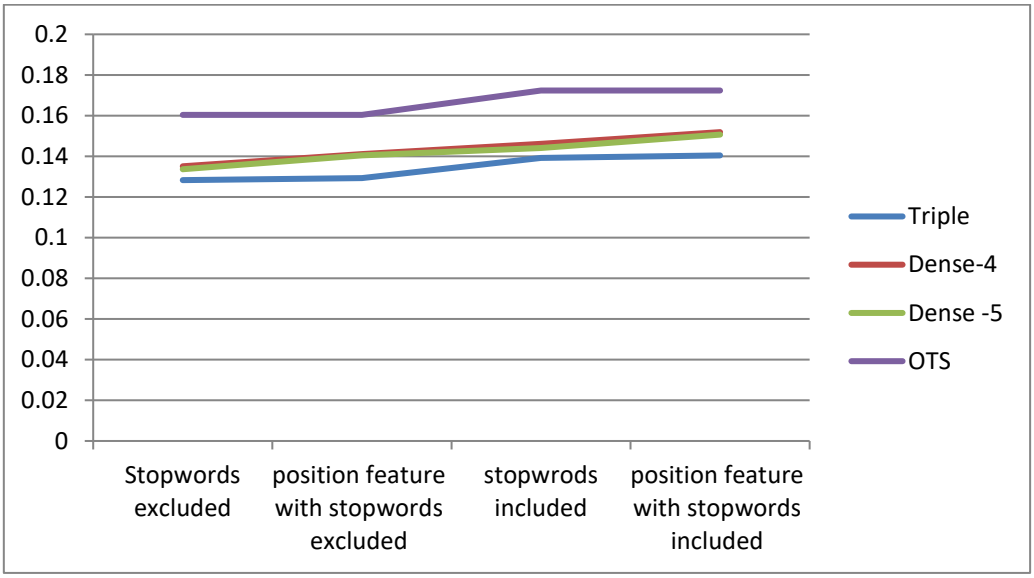

Figure 4.7: Rouge--2 score vs. different rouge settings and features on DUC-2001 corpus.

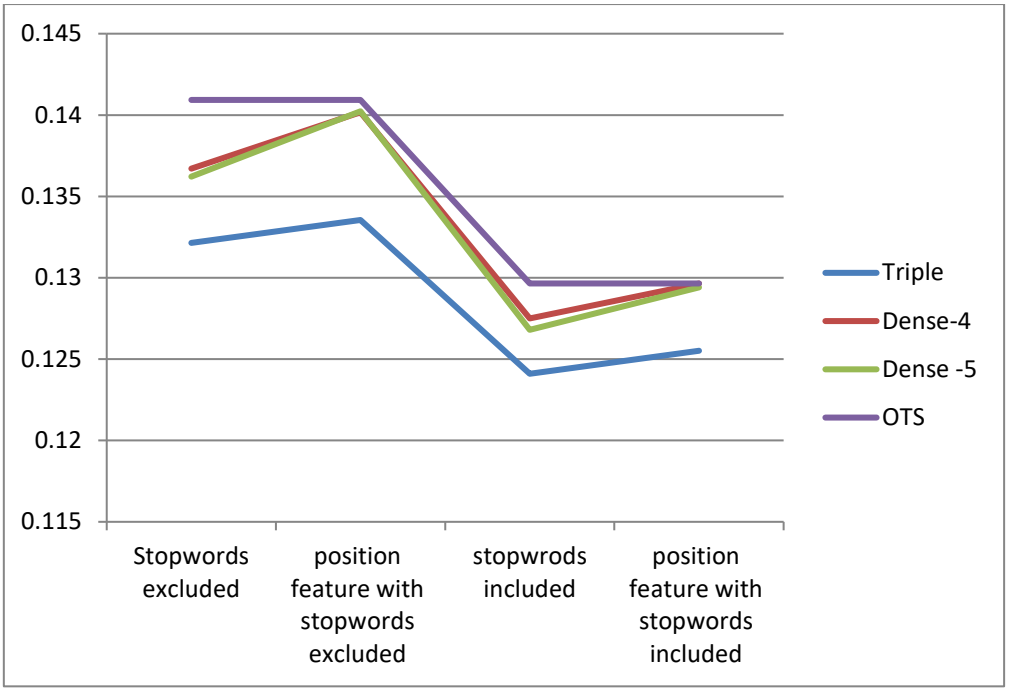

Figure 4.8: Rouge--W score vs. different rouge settings and features on DUC-2001 corpus.

Table 4.3 and Table 4.4 shows the scores for experiments on DUC-2002 corpus with two ROUGE evaluation settings *stop words included* and *stop words excluded*. In Figure 4.9, Figure 4.10 and Figure 4.11 we have plotted the Rouge-1, Rouge-2 and Rouge-W score of *Triple based summariser*, *Dense semantic Graph-4 summariser*, *Dense semantic Graph-5 summariser* and *OTS summariser* vs. different rouge setting on DUC-2002 corpus. ROUGE scores improves on the shortest dependency based graph, until the distance 5. During result analysis we have observed that ROUGE score decreases or becomes approximately constant if we increase the distance over 5.

Including sentence position as a feature significantly improves the results in summarisation from triple based graph. However including position as a feature does not improve results much for the shortest dependency distance path based dense semantic graph. Hence we can conclude that ranking of dense semantic was already giving accurate results and thus this graph will be useful for non-news domain data as well where the positon in the sentence does

not have much impact on indication of its importance. DUC-2001 and DUC-2002 corpuses are news data and sentence position has great impact on important information extraction in news data, as important information is conveyed in the initial sentences of news. Overall the shortest distance based semantic graph performs better in ranking the sentences and is comparable to the benchmark system OTS.

Table 4.3: ROUGE-Scores on DUC-2002 corpus with ROUGE setting - stop words included.

System	Rouge-1	Rouge-2	Rouge-W
Triple Semantic Graph	0.40071	0.15798	0.12955
Dense semantic Graph -2	0.43214	0.17234	0.13948
Dense semantic Graph -3	0.43542	0.17497	0.14081
Dense semantic Graph -4	0.43646	0.17552	0.14076
Dense semantic Graph -5	0.43633	0.17659	0.14103
Triple Semantic Graph + position	0.42836	0.17061	0.13855
Dense semantic Graph -4 + position	0.43646	0.17553	0.14076
Dense semantic Graph -5 + position	0.43658	0.17712	0.14115
OTS	0.45107	0.20317	0.13904

Table 4.4: ROUGE-Scores on DUC-2002 corpus with ROUGE setting - stop words removed.

Summarisation System	Rouge-1	Rouge-2	Rouge-W
Triple Semantic Graph	0.33864	0.14714	0.14143
Dense semantic Graph -2	0.37154	0.16221	0.15465
Dense semantic Graph -3	0.37494	0.16409	0.1563
Dense semantic Graph -4	0.37666	0.16498	0.15694
Dense semantic Graph -5	0.37919	0.168	0.15778
Triplet Semantic graph +position	0.36465	0.16016	0.15231
Dense semantic Graph -4 + position	0.37666	0.16498	0.15694
Dense semantic Graph-5 + position	0.37937	0.16846	0.15793
OTS	0.38864	0.18966	0.15766

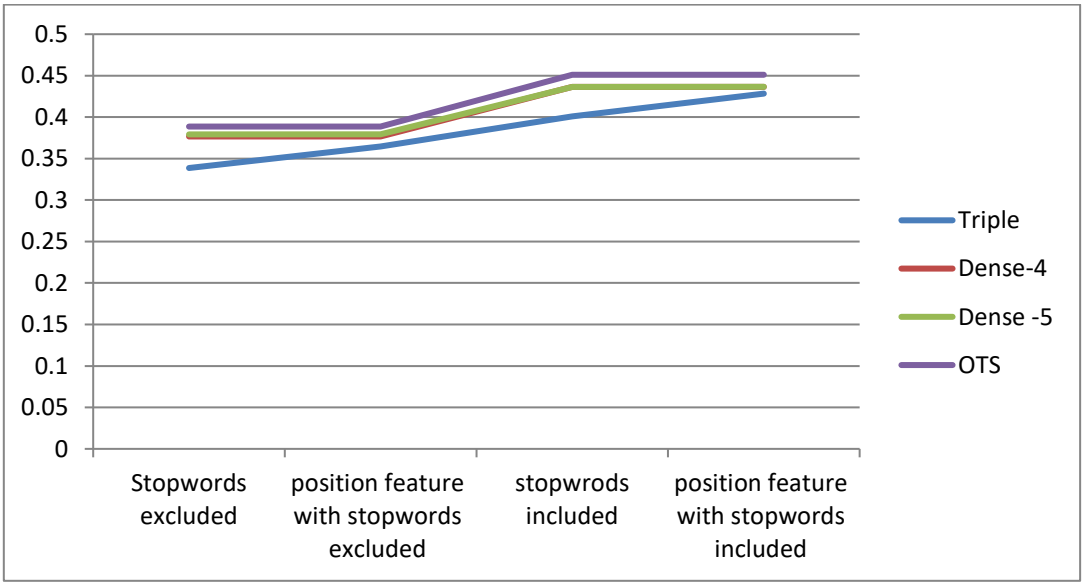

Figure 4.9: Rouge-1 score vs. different rouge settings and features on DUC2002 corpus.

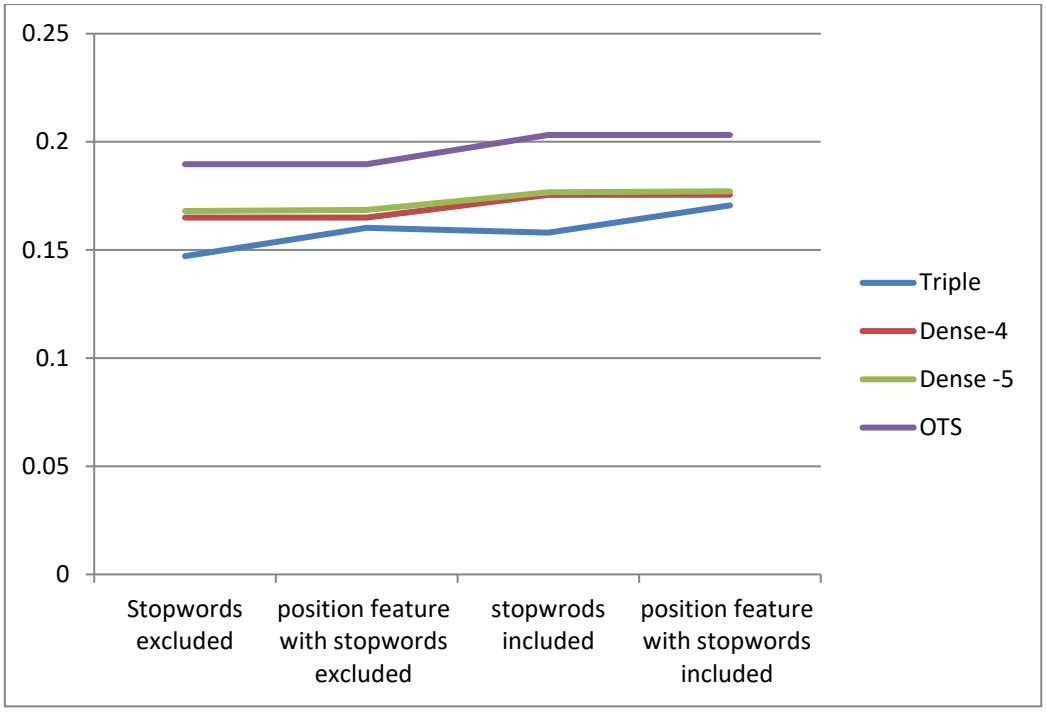

Figure 4.10: Rouge-2 score vs. different rouge settings and features on DUC2002 corpus.

(b)

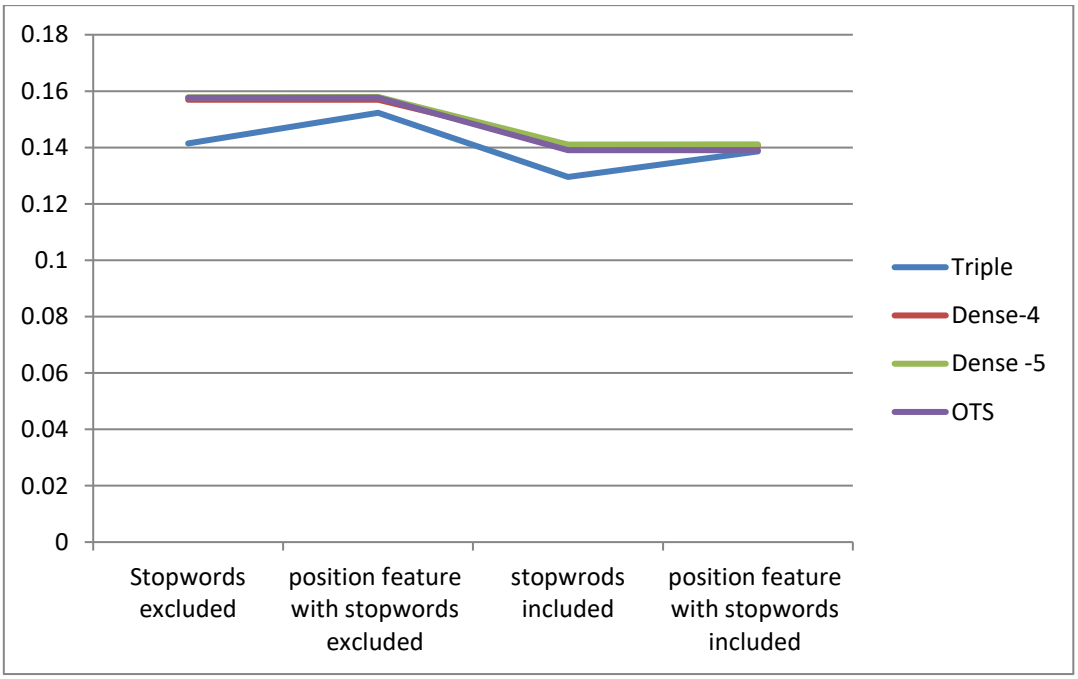

Figure 4.11: Rouge-W score vs. different rouge settings and features on DUC2002 corpus.

4.7 Dereferencing

In our summarisation approach to generate the better connected semantic graph we were pre-processing text for replacing all pronominal references with the exact named entity. In addition, shorter references such as last names were being replaced with full names if they exist in the text. To make summary more coherent we were extracting sentences from this pre-processed text because If sentences are extracted from original text then dangling references issues occurs in the generated summary. This happens because references to named entities are lost when sentences with pronouns are extracted. After analysing summaries generated from our system we have found that sometimes a reference is being repeated in same sentences many times because the original sentence had many pronominal references to it.

If the main named entity has a lengthy name then repeated references to it blocks space for more information as the summary is limited to 100 words. It also makes the summary unreadable. If we extract the original sentence, we may get more informative words in the 100-word length summary but we will face issues of dangling references. To find an optimal solution for this problem we have reviewed the literature on how references to named entities are made [190]. A general trend was observed of modifiers used for named entities differing with the position of these mentions in the sentence. The first mentions use most of the modifiers for the named entity and then later mentions in the sentence/sentences use part of the first mention which could be either the last name or first name. Pre-modifiers and post-modifiers are not likely to be used together for any mention. It was also seen from probabilistic analysis of Markov chains made from noun phrase data, that once a non-modified mention has been used, it is unlikely that any other mention of the same named entity will add modifiers to it. Apposition is the most probable post modifier to be used at

first mention of entity, rather than prepositional or relative clauses [191]. For pronoun generation, basic rules are set so that only if a main entity reference exists previously and there is no other similar gender entity reference exists in the same sentence can a pronoun be used as reference.

Following the statistical results achieved by Markov chain realization of noun phrases and pronoun generation rules, to keep summary readable and coherent we came up with few changes to the summary extraction. We still extract summary sentences from the prep-processed text after replacing all co-references. Now since every occurrence has the exact reference, we go through each sentence and then keep only the first mention of the entity and replace later references with pronominal references where feasible. We call this process de-referencing. We have performed summarisation with de-referencing with our best performing system *dense semantic graph-5* without any other feature with ROUGE setting *Stemmed words and stop words included* on DUC-2002 corpus and found that results improved in 2-gram and longest sequence matching. Results are shown in Table 4.5.

Table 4.5: Improved results with dereferencing.

Summarisation System	Rouge-1	Rouge-2	Rouge-W
Dense semantic Graph -5	0.43633	0.17659	0.14103
Dense semantic Graph -5 with de-referencing	0.43640	0.18111	0.14235

To analyse the summaries generated from dense semantic graph based text summarisation approach we extracted the best 10 summary sentences of the documents. We started

comparing these sentences to human written reference summaries that comes in DUC data sets. Basically reference summaries contains two human written summaries which are around 10 sentences length in all, some of which may have common sentences between them. We observed that in the system generated summaries sentences were quiet longer than the sentences found in reference summaries. Reference summaries are abstractive summaries written by humans and may contain words which are not part of the original document. From comparison of sentences which provides the same information in the system generated summary and the reference summary, we could see that the sentence in the reference summary contains only a part of the original sentence in the document. Most of the subordinate clauses are removed from the original sentence when quoting that information in reference summary. In this way, the summary also contains more diverse information rather that one large sentence occupying a bigger part of summary. Also the literature suggests that in earlier approaches adjectival clauses of the sentences were removed based on their relevance and importance to be included in the summary. It indicates to achieve the quality of summaries produced by humans, summarisation systems have to go beyond pure extractive summaries. There should be systematic approaches to divide the information into smaller units and later combine them into useful summary. This has led to our third research study described in next Chapter.

4.8 Conclusion

In this Chapter we have researched two ways to construct semantic graph of textual information; *the triple based semantic graph* and *the dense semantic graph.*

We have also implemented two different summarisation systems by incorporating the ranking information from these graphs. Our summarisation system based on *triple based semantic*

graph differs from earlier summarisation approach which unlike our approach uses *PageRank* score as one feature among many for supervised text summarisation. Our both summarisation approaches are unsupervised. In the second summarisation method we have utilised a novel semantic graph *dense semantic graph* made from long distance dependency relations. In evaluation Section we saw that summarisation results from *dense semantic graph* exceeds the summarisation results from *triple based semantic graph*. Although the performance of *dense semantic graph* based summariser does not exceed the benchmark results, it is comparable.

The analysis of results confirms the assumed hypothesis that the more dependency relations included in semantic graph generation the more accurate are the *PageRank* scores of semantic nodes and thus the more accurate are the rankings of the sentences for text summarisation. When extra feature sentence position was included, it improved the ROUGE scores for summarisation. This shows if we will consider more features on the new graph, further improvements can be made to text summarisation.

In future work on this, semantic similarity measure and word sense disambiguation can be applied to improve the connectivity in the *dense semantic graph* by identifying more relations between nodes. Dereferencing techniques could be research further to include appropriate references of entities based on gender and context. Efficient dereferencing will make summaries more readable and provide space for more summary content to be included. Additionally the dense semantic graph can be improved to be more visually perceivable and more efficient for direct abstractive summary generation from it instead of extracting from original document.

Chapter 5

Object Oriented Semantic Graph

In Chapter 4 we described two semantic representations -- triplet based semantic graph and dense semantic graph. In both representations graphs were generated from textual information by analysing the dependency relations of words in sentences. We have seen that considering more dependency relations as done in dense semantic graph provides better structure for summarisation and thus improves the results.

However the dense semantic graph and other graph representations used for summarisation are based on surface level terms. Each node in these graphs is atomic and cannot be converted to summary directly. As described in Chapter 4 this kind of graphical representation of text is good for ranking n-grams from text and can be used in term ranking based extractive summarisations. But this graph cannot be converted to summary itself, since it lacks the semantic relation to join terms and form sentences. The main aim of our research is abstractive summarisation and for that we need a deeper semantic representation of document.

Text documents can be visualised as interaction of objects. Object can be a person, a place or any entity being talked about in the document. These objects have certain properties and specific behaviour that gets described in the text. Their interaction with each other forms the relations between them. This inspired us to include Object-Oriented analysis and design (OOA/D) principles while designing semantic representation of document. OOA is the analysis of a problem with concentration on the domain concepts which are considered as domain objects and OOD is the design of software solutions by emphasizing on domain objects and their interaction and associations. [192]

OOA principle promotes encapsulation of behaviour and properties of each object as a class. Through inheritance subclasses inherit all properties and methods from derived classes. Although it is not possible to design the representation for natural language which complies with all OOA principles due to the ambiguity and complexity in it, but our work is a first step. Previous work on modelling text for Object-Oriented graph generation has been discussed in next Section.

5.1 Modelling natural language text

Previous work to generate an object–oriented graph representation of text document has been done for requirement analysis phase in software development lifecycle. Initially Abbott [193] gave a set of rules to identify objects/classes from nouns and relations from verbs. Later by extending Abbott's theory, researchers [194,195]presented semi-automatic and automatic systems to generate object-oriented graphs of requirements documents written in a controlled natural language (CNL) text. In CNL sentences are made up of restricted predefined set of vocabulary and often restricted to be more action oriented rather than description oriented.

Latest work by Elbendak [196] and Vidhu Bhala [197] demonstrates the recent development in this field. Elbendak made a semi-automatic system to generate the *Unified Modelling Language* (UML) model of requirements given in natural language text. They analysed part of speech tags and phrasal parse structures of the sentences. They also took user inputs in few cases to build the UML diagrams of the requirements presented in natural language. Vidhu Bhala's work [197] was inspired by Elbendak's work [196], and they gave an extensive description to implement the theory of object/relation identification in automatic system. Their system works by analysing the dependency relations between words in the sentences and by using the part of speech information of every word. Their work is closely related to

how we are proposing to extend our dense semantic graph. Most of the work on object–oriented graph derivation from text including previous described approaches has been done for functional requirement specifications. We have worked towards generating object-oriented graph for any unrestrained natural language text.

Natural language text is an amalgamation of functional and non-functional text. Functional requirement poses a restriction that every sentence should describe some action performed in the system. It does not include any sentence that describes or restrains the way system does the work in terms of quality or other measurable requirements. One example of functional requirement is the sentence "*This system adds value to binary tree and searches for value from binary tree*". Whereas the non-functional requirement sentence about same system is "*System should be fast*." Functional text is relatively easy to model in terms of classes and relations, but non-functional text often needs more analysis to identify appropriate classes and properties of classes from them.

5.2 Related semantic graphs

Other than modelling requirement specifications the work similar to the proposed work are semantic graphs which have focussed on concepts in the text, one of which is concept graph. It is a connected graph of concepts described in detail in Section 2.3.3 of Chapter 2. To map the text document to concept graph, concepts in the document are identified, which according to basic definition given by John F. Sowa [198] are action nouns and action verbs in the text, and these concepts are connected by thematic relations defined for that action. Thematic relations are the conceptual relation between process and the participating concepts i.e. *agent, theme, experiencer, medium*. *Agent* relation describes the doer of process, *experiencer* relation connects the process to its receiving participant and *medium* describes the relation of

process to the resource utilised for its completion i.e. in sentence *Sam informed Mary on phone.* the concepts connected with thematic relations are: *Sam <-agent> told, told<experiencer-> Mary, told<medium->phone.* Concept graph also considers negation and conditional clauses and supports reasoning over them. For automated processing concept graphs can be converted to *conceptual graph interchange format (CGIF)* format or *knowledge interchange format (KIF)* format[199]. Some work has been done to automatically convert text to concept graphs[200,201], and the use of it in domains such as patent claim processing, medical document mining [200,202,203] using linguistic resources.

Another extended and interchangeable approach is *resource description framework* (RDF) triples, which is the building block of semantic web. In RDF, everything is a resource and connected through different attributes i.e. rdf:property, rdf:subclassof. RDF schema and its extension *web ontology language* (OWL) are well developed and supported by W3C [204]. RDF triples has been used for information extraction, linking open data in Wikipedia to ontologies [205,206]. Another semantic network based representation language is *universal networking language* (UNL), which has been used in machine translation [207,208]. UNL is made of three types of nodes, universal words, universal relations and universal attributes. Apart from these popular notations-UNL, RDF variations of similar graphs such as event graphs [152]which are made of relations between event mentions in the sentences and semantic graphs[149,209] such as logical triple based graph discussed in detail in previous Chapter have been used for information retrieval and relation extraction tasks. Common processes in these semantic representations include (1) processing text to divide it into single units, and (2) identifying relation between these single units, and (3) mapping them to ontology.

Although not all text in natural language can be mapped to these notations, there are tools in computational linguistic to handle the complexity of texts. The information units and their

structure in semantic representations so far looks suitable for populating a knowledgebase or GraphDB but it looks far from how a reader would like to see a document being interpreted by a computing machine. Every text document, other than the domain of numerical data, is a collection of interacting entities/objects. Every entity has its own set of properties, behaviour and its details; and it is related to other entities by some relation. As per the design of previous semantic graph approaches a proper ontology will be required to make sense of graph nodes as property or behaviour of entities for further processing tasks. Ontologies are domain specific and may not be available in all cases.

A thought provoking question is whether the nodes of a semantic graph should be made of composite units instead of atomic units. Visualization of information in terms of composite units is a better way to design the semantic graph representation of a text document because in real world scenario entity is expressed with bigger text descriptions and many modifiers are attached to it. Here we bring the object-oriented analysis and design principles to add the real world feel to the semantic graphs and propose an object-oriented semantic graph (O-O semantic graph) to represent knowledge and information in a text document. In this graph we expect to see object, their interaction, their details and it should be self-explaining for computation purposes in case of lack of ontologies.

Here first we describe the general structure of a sentence and then we discuss the issues faced when we model natural language text due to the structural differences from functional text and later we describe how these issues are dealt during construction of O-O semantic graph. From here onwards we will differentiate the object of O-O semantic graph from the object of a verb by referring the later by v-object.

5.3 Sentence structure

Sentences are made up of noun phrases (NP) and verb phrases (VP). Noun phrase consists of a head noun and optional pre or post modifier phrases. These modifier phrases are adjective phrases, participle phrases, prepositional phrases or subordinate clauses.

Verb phrases consist of a verb group and its complement. Verb group consists of a lexical verb and optional auxiliary modifier verbs. Complement of a verb group can be noun phrases or subordinate clauses. These complements of the verb form the object of verbs; direct v-object, indirect v-object and prepositional v-object. This complementation decides the type of verbs. Verb phrase also includes optional prepositional phrase as modifier. These modifiers can be divided into adjunct adverbial, disjunct adverbial or conjunct adverbial [210].

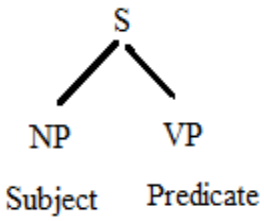

Figure 5.1: General sentence structure.

Subordinate clauses have structure similar to sentence. They can be categorized into finite, non-finite and wh-clauses. Finite clauses contains proper forms of verb with tense information, whereas non-finite clauses contain verbs in four forms: 1. bare form (i.e. *She made him darn her socks*), 2. *to-verb* form (i.e. *He is thought to be hiding in Brazil*), 3. passive participle verb form (i.e. *the palanquin loaded, we took a rest*) and 4. *verb-ing* form (i.e. *Getting up before dawn was not that good*). Non-finite clauses has some missing

syntactic constituent (mostly subject) and can be determined from main clause. Wh-clauses has connecting words from wh-words (what, where, which, who, when).

5.4 Complexity involved in modelling natural language sentences

In semantic form, we describe the top level noun phrase of sentence as subject and top level verb phrase as predicate. Subject is the main element talked about in the sentence and predicate consists of event described about subject, and the v-objects of event. Modifier is another semantic unit which consists of descriptive words that restrict the described entity to identifiable instances.

Previous approaches on modelling natural language text have focused on functional sentences where all semantic entities (subject, v-object and modifiers) can be drilled down to one syntactic unit (i.e. a compound noun/noun, verb, adjective). It has not considered cases where subordinate clauses can itself function as subject, v-objects modifiers or predicates. To deal with such complexities of natural language more grammatical and semantic relations have to be considered to represent the complete information in a model. Here we discuss the limitations in more detail.

5.4.1 Complex sentence structure

As we can see from the sentence structure described above, natural language sentences may contain sentences within them, which are called subordinate clauses. These clauses may be complete sentences or may have one or more syntactic element missing (mostly subject). It can have various roles relative to other syntactic units in the sentence, i.e. subject of the verb,

complement of verb or modifier of noun or adjective. In modelling approaches based on syntax parsing and dependency parsing the clauses were not considered for subject/v-object extraction or properties extraction for object/relation. Methods differ to identify subject from finite (‘*that*’) and non-finite clauses complement. In our proposed approach we have considered patterns of dependency relations to identify subject/v-object/predicate from the clauses.

5.4.2 Modal sentence

A modal sentence expresses the possibility of some event depending on certain conditions. This is expressed by additional auxiliary verbs along with the main lexical verb in the verb group. A few example sentences are:

i. *He should be here by now.*

ii. *I could swim quite well when I was younger.*

In functional text, every sentence describes some action, so either it does not have modal verb or the model verb used there is ‘must’ or ‘should’.

5.4.3 Negation of verbs

It is common to have sentences with negation of verbs. This information has been ignored in previous work to model requirement specifications.

5.4.4 Multiple references of same entity

In natural language it is common to refer to the same entity by different references which are not pronominal. It could be name, type or adjectival clauses i.e. given the text snippet: *David*

Beckham was spotted in Harrogate yesterday. The 38-year-old footballer was there to enjoy tour de France, we have one non-pronominal reference to the named entity *David Beckham which* is *38-year-old footballer.* In these cases of non-pronominal co-references none of the reference can be replaced by other reference completely, as it will lead to information loss. In earlier approaches of modelling text document it was assumed that all references have been resolved to one phrase, which may not be possible for natural language text. But identifying co-reference relation between them can help in connecting the corresponding objects in the object-oriented semantic graph.

5.4.5 Inherent semantic knowledge

Entities may have relation to each other which are not explicit in the text. These could be ontological relations, which can be acquired from knowledgebase. Ex. '*Falcon'* and '*the bird*' can be two references to same entity. When these cannot be resolved by co-reference resolver, we can connect the two references by ontological relation '*isA*' derived from a knowledgebase by looking for hypernym relation between words.

5.4.6 Properties of nouns from post modifying phrases

In addition to adjectives and clauses, prepositional phrases can also modify nouns. Again modification can be of two type; pre-modification and post-modification. Although we have not done work to resolve this issue in our work, it can be one good area to explore.

In this Section we have described the structure of natural language text and the limitations in generating object-oriented graph from it. In the next Section we describe the proposed rules to generate object-oriented semantic graph of natural language text to fulfil the gaps discussed in this Section.

5.5 Proposed O-O semantic graph

As described earlier modelling natural language is difficult due to complexities discussed in Section 5.4. We have proposed here a systematic approach to handle complexities of natural language to generate an O-O semantic graph of the information present in text. O-O semantic graph of a text document is *G= {O, R}* where *O* is set of objects and *R* is set of relations identified from that text document. Objects should represent the core information units around which the complete narration of text document revolves. Generally in other information extraction approaches main informative units of the document are determined based on surface level features such as frequent terms, cue words or title words. Instead of only relying on surface level features we devise semantic methods to identify it.

Second component of O-O semantic graph is relations between objects, which depicts the interaction activities between important units of text. Sub-components of object are properties and operations of object, which enrich and refine the information covered about object and helps in resolving ambiguities. We introduced a new feature, which is the sub-component of relation: property of relation between objects. It is different from the previous approaches and is necessary for natural language text, because the relationships between objects may depend on certain conditions. Property of a relation accommodates the information about verbs (i.e. negation, modality) in the O-O semantic graph. In earlier literature, verbs are considered to be used in only affirmative sense. But modal sentences as discussed in Section 5.4.2 add different possibility of occurrence to the verb. The following rules have been taken from literature of generating object oriented graph of requirements specification documents.

Elbendak [196] defined following rules:(1) All nouns are candidate classes, (2) All verbs are either candidate operations or candidate relations.

Following the above rules and to avoid unnecessary classes to be added to semantic graph Vidhu Bhala [197] proposed that nouns that appear as subject are surely classes, but nouns that only occur as v-objects may or may not be classes. Both of the above stated systems tries to general UML diagrams of classes, their capabilities (attributes and operations) and their relation with other classes. In the research works presented in current Chapter new rules have been formed in addition to these rules, by analysing text and their manual summaries.

In the remainder of this Section we first discuss the proposed rules, and then describe how it is implemented from dependency parse of text.

5.5.1 Rules to identify objects

Rule 5.5.1.1: All named entities which are location, name of person, name of organization are made objects in the graph. This is a new rule to identify objects.

Rule 5.5.1.2: All nouns which act as a subject of verb, in the triple *subject->verb->v-object* are taken as object in the graph. We bring novelty in this old rule by extracting subject from clauses. Pronouns are resolved to referring nouns using co-reference resolver.

Rule 5.5.1.3: All nouns which are direct v-objects of verbs or are prepositional v-objects of verbs may be objects in the graph if they act as subject of some verb. We modify this old rule to include frequent v-objects of verbs even if they are not subject of any verb. Frequency of v-object is determined by comparing the count of its occurrence in all possible triples to a predefined threshold value.

Generally subject of verbs are considered worthy of being projected as an object in the O-O semantic graph, but here we modify this rule to adjust the story writing style of descriptive texts. In descriptive text a lot of information is written about an entity (i.e. a person) that may

never be subject of any verb (i.e. performer of any action). But frequency of its occurrence indicates that entity to be important enough to be projected as an object.

5.5.2 Rules to identify relations between objects

Relation between objects can be identified from these rules.

Rule 5.5.2.1: If two objects are in *subject->verb->v-object* triple from previous identification of subjects and v-objects then this verb becomes a connecting relation between them. This rule has been applied in earlier approaches as well.

Rule 5.5.2.2: If two objects are in *subject->verb->propositionally connected word* triple then this verb with the prepositional post modifier becomes a connecting relation. This rule has been applied in earlier approaches as well.

Rule 5.5.2.3: As described in Section 5.4.4 sometimes same entity may be referred by two different words, which are not pronominal references. In this case if these connections are identifiable from co-reference resolver then it forms a relation between two indirect references to the same entity. We generally merge them into one object and make one the property of other. It is a new rule added here.

Rule 5.5.2.4: We propose to use ontologies (i.e. WordNet) to identify existence of a hyponymy or synonymy relations between objects in the object-oriented graph. This is a new rule added here to fulfil the gaps stated in Section 2.2.4.

5.5.3 Rules to identify properties of objects

Object has certain behavioural properties, which can be identified from connecting modifier words, as described here.

Rule 5.5.3.1: Adjectives represent additional information provided about the object, which can differentiate this object from other objects of same class. All adjectives are added as property of the corresponding objects.

Rule 5.5.3.2: Apostrophe gives additional information when it connects two nouns.

Rule 5.5.3.3: Prepositional phrases which are connected to nouns works as post modifiers as described in 2.1. We extract information about object from it.

Rule 5.5.3.4: If one noun acts as subject of other noun in dependency relation, then the other noun is considered as a property of the subject noun. Ex. in "*Mr. Clinton was the president of United States*", *president* is the property of *Clinton*.

5.5.4 Rules to identify operations performed on/by object

Rule 5.5.4.1: All verbs connecting this object to non-object nouns are considered as some event done by this object. They are sequenced according to their occurrence in the text, as an operation performed on/by this object.

5.5.5 Rules to identify properties of relations

This is a novel attribute added to the O-O semantic graph. There are two rules to identify properties of relations.

Rule 5.5.5.1: As stated earlier, natural language contains modal sentences, which expresses the possibility of event in different ways. Here we extract the modal information of verb and attach it as a property to the relation.

Rule 5.5.5.2: If there is a clausal dependency between two verbs then we add second verb as a property to the relation derived from the first verb.

5.6 Development of O-O semantic graph

All rules of Section 5.5 are implemented to test the efficiency of O-O semantic graphs in natural language processing tasks. Figure 5.2 shows the pipeline of tasks involved in object-oriented semantic graph generation. We have utilised dependency parse structure of sentences to achieve the desired semantic graph.

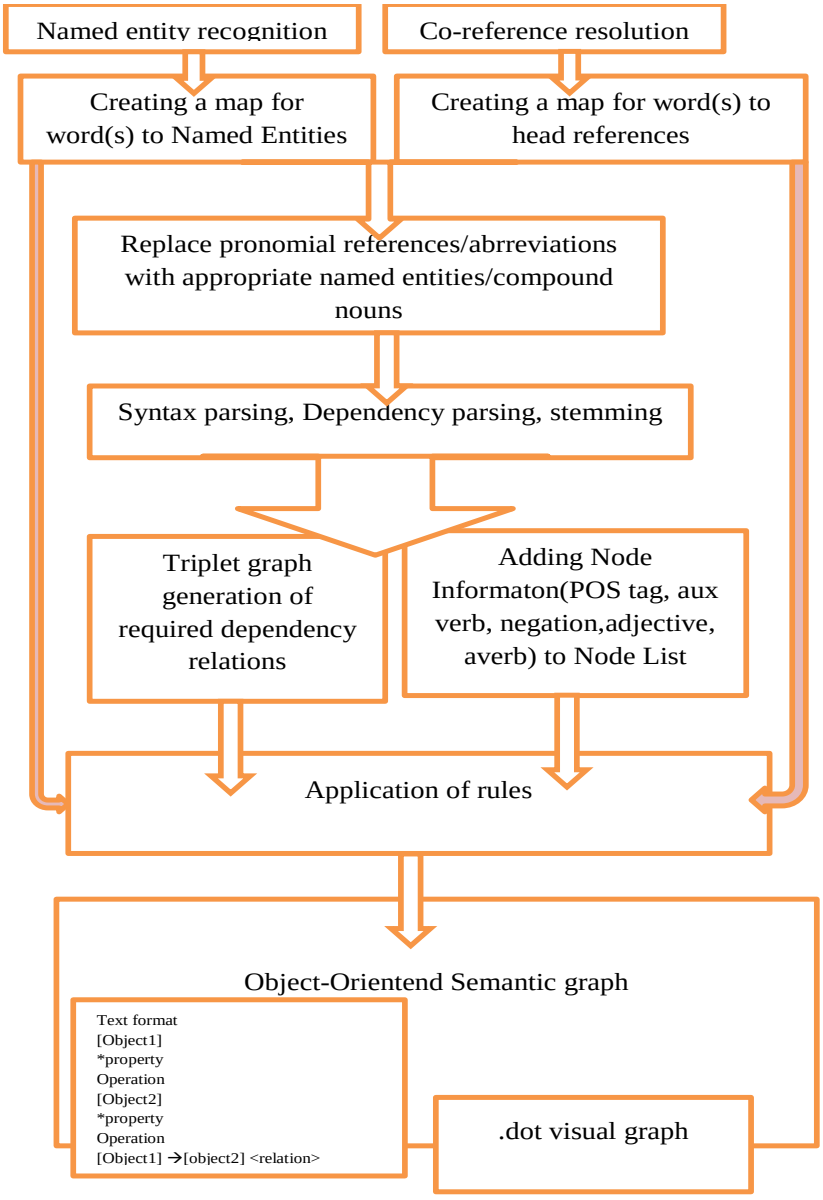

Figure 5.2: Pipeline to generate object oriented semantic graph.

5.6.1 Pre-processing of text

Pre-processing of text involves named entity recognition and resolving co-references. Co-reference resolution and replacement steps remain same as done for previous study on dense semantic graph described in Section 4.2.1 of Chapter 4. However to utilise the connections further even after references have been replaced, we construct a map *wordToMentionMap* of word spans to unique reference numbers. We have utilised Stanford's CoreNLP toolkit[211] for co-reference resolution. In Stanford's co-reference mention list all mentions to same reference along with head mention are clubbed together as a list. In this mention list references can be group of words with modifiers. We break down the bigger reference to minimum word span and create word to mention map. This map is used in O-O semantic graph to decide relations between objects based on co-reference. While creating *wordToMentionMap* we ignore a bigger reference in mention list if its constituents are also separate references pointing to same head mention. For example a co-reference mention list by Stanford resolver is *["Hurricane Gilbert , packing 110 mph winds and torrential rain ," in sentence 1, "Hurricane Gilbert" in sentence 1, "Hurricane Gilbert" in sentence 24]*.It contains 3 mentions to same entity. In this example we ignore the first mention *"Hurricane Gilbert, packing 110 mph winds and torrential rain,"* due to two reasons. First reason is that second mention *"Hurricane Gilbert"* is the constituent of first mention and both refers to same entity and second reason is that second mention makes better succinct reference while avoiding the other modifier information about speed of the wind in *Hurricane Gilbert*. In case when a reference number has been assigned by the bigger reference in the *wordToMentionMap* and then later a different reference is identified for its smaller subpart then this smaller sequence of words are changed to point to new reference leaving same the remaining ones. This is the case of overlapping references. Yet there are complexities

involved in few cases to provide a mapping from word to mention. So we have come up with an optimum solution by analyzing different cases and this solution has been explained here through the pseudocode in Figure 5.3(i), 5.3(ii).

```
Initialize mention_map to Hashmap<Integer sentencenumber, SortedHashMap<Integer wordnum, Integer MentionId> wordToMentionMap>
In mention_map fill sortedHashmap for each sentence with null values.
Initialize currentMentionId=1;
For each metionList from the coreference list
        For each Mention in mentionList in textual order
        Only if this mention is not divided into further references in this mentionList//Line A
                sentenceNum=mention.getSentenceNum;
                StartWordNum=mention.getStartWordNum;
                EndWordNum= mention.getEndWordNum
                ExistingMentionId=mention_map
[sentenceNum].wordToMentionMap[StartWordNum];
                Initialize begin and end : begin=-1, end=-1;
                If Existing MentionId is NOT NULL //Line B
                        {//we try to make decision about whether existing mentionId covers longer wordspans than current mention id
                        Find the surrounding sequence of words with same existing mention id that covers the StartWordNum
                        Beginning of this sequence is marked by leftmost word found without any gap
                        Ending of this sequence is searched till the last rightmost word added in the mentionList without any gaps
                        Assign beginning and ending numbers to begin and end
                          If startWordNum>=begin and EndWordNum<=end
                                For each word W from StartWordnum to EndWordNum
                                replace mentionId: wordToMentionMap.add(W, currentMentionId);
```

Figure 5.3(i): Pseudocode to generate wordToMention Map from Co-references.

```
else
        for each word W from end to toEndWordNum
            if wordToMentionMap[W]=null
                replace mentionId: wordToMentionMap.add(W,
                    currentMentionId);
            else
                if WordLength(headMention(wordToMentionMap[W]) >
                    wordLength(CurrentMention)
                replace mentionId: wordToMentionMap.add(W,
                    currentMentionId);//Line C

else //when ExistingMentionId on StartWordNum is NULL
        for  each word W  from StartWordnum to EndWordNum
        if wordToMentionMap[W]=null
            replace mentionId:   wordToMentionMap.add(W,
                currentMentionId);
            CurrentMentionId= CurrentMentionId+1;
```

Figure 5.3(ii): Pseudocode to generate wordToMention Map from Co-references.

For the text "*We should know within about 72 hours whether it's going to be a major threat to the United States," said Martin Nelson, another meteorologist at the centre*." The recognized mentions are *["Martin Nelson, another meteorologist at the center" in sentence 26, "Martin Nelson" in sentence 26]*. This depicts the case of similar reference by bigger and smaller wordspans. Code Line A in pseudocode in Fig 5.3(i) ignores the bigger first mention *"Martin…..center"* , and makes word span of second mention *"Martin Nelson"* to point to the mention id decided for it.

Following are three different overlapping references encountered in order.

1. "*director of the National Hurricane Center in Miami*" in sentence 4
2. "*Miami*" in sentence 4
3. "*the National Hurricane Center in Miami*" in sentence 4,

Table 5.1 shows the change of reference ids of words after encounter of each overlapping mentions by code lines B to C in pseudocode explained in Figure 5.3.

Table 5.1: Changes in mapping of words to reference ids after encountering each mention.

	1st mention	2nd mention	3rd mention
Director	1	1	1
Of	1	1	1
The	1	1	3
National	1	1	3
Hurricane	1	1	3
Center	1	1	3
In	1	1	1
Miami	1	2	2

After creating map for word numbers to reference numbers, another map is created for word numbers to recognised named Entities. This step remains same as done for dense semantic graph generation in Chapter 4. Next pre-processed text is parsed using syntax parser and dependency parser. We construct a node list to store feature information about nodes to be used later. Following features are selected for further use to construct the O-O semantic graph.

1. Part of speech tag

2. Auxiliary verb if node is verb

3. Negation

4. Connected Adjectives

5. Connected Adverbs

6. Original word without stemming

7. Referenced mention id

8. Connected nouns by preposition

5.6.2 Object identification

As shown in the pipeline to generate O-O semantic graph in Figure 5.2 the input for the generation of O-O semantic graph is dependency parse of the sentences , *word_to_mention* map and *word_to_named entity* map. We implement rule 5.5.1.1 by going through the *word_to_named entity* map and add each key of this map to list of objects if the type of entity belongs to categories *organization*, *person* or *place*. Each predicate that was used in identifying the object is stored for further processing in the form of *tuple (object, rel)* in *ObjectMap*.

For sentence *Barrack Obama is the President of United States,* two objects are added to semantic graph using rule 5.5.1.1: *(i) Barrack Obama (ii).United States* because *"Barrack Obama"* falls in named entity category *person* and *"United States"* in category *place.*

Rule 5.5.1.2 is implemented by identifying all possible subjects of predicates from clauses using *xsubj*, *rcmod*, *partmod* dependency relations and using *nsubj*, *nusbjpass* relation from sentences. Each extraction of object from predicates is implemented as follows.

i. add X to ObjectList if $nusbj(\mathrm{Y}, \mathrm{X}) \in dependecyRelationList$ *and* $pos(X) = \text{'NN'}$

ex. *Mitchell called reporters to cover the event.* Here rule matches dependency relation *nsubj(called, Mitchell)* and Mitchell becomes an object in the graph.

ii. add X to ObjectList if $\exists$ agent(Y, X) $\in dependecyRelationList$ *and* $pos(X) = '\text{NN}'$

ex. *Cuba was devastated by the storm.* Here rule matches dependency relation *agent(devastated, storm)* and *storm* becomes an object in the graph.

iii.*a*dd X to ObjectList if $\exists$ nusbjpass(Y, X) $\in dependecyRelationList$ *and* $\nexists agent(Y, ?)$

and $pos(X) = '\text{NN}'$

ex. In the previous sentence dependency relation *nsubjpass(devastated, Cuba)* did not contribute any object, since there existed doer agent identified by relation *agent(devastated, storm).* However, in a sentence with only passive subject i.e. *Storm approached from the southeast with sustained winds of 75 mph.* the subject *Storm* becomes an object of the graph.

iv. add X to ObjectList if $\exists$ xsubj(Y, X) $\in dependecyRelationList$ *and* $pos(X) = '\text{NN}'$

ex. *Tom likes to eat fish*. From dependency relation *xsubj(eat,Tom)* we identify object *Tom*. In such cases of open clausal complements we combine both verbs to form a single relation which is explained in next Section.

v.add X to ObjectList if $\exists$ rcmod(Y, X), nsubj(Y, wh − noun) $\in dependecyRelationList$

and $pos(X) = '\text{NN}'$

Dependency relation *rcmod* indicates relative clause modifier of noun. Above rule resolves the references from relative clause modifier nouns. Ex. *Cook* from the sentence *Bill saw the cook who made the cake other day.*

vi. Appositions are converted to *nsubj* relation of two nouns and earlier rules are applied to extract object.

Fragment *Obama, President of America* gets converted to *Obama is President of America.*

vii. Owner of possess relations are converted to Objects.

viii. Frequent verb-objects are tracked to add as Object. A detailed description of this is given in the next Section.

All prospective object words are converted to compound nouns or to referring named entity. All objects are differentiated by unique headword and sentence numbers and only if new object maps to same reference it is merged to similar object.

5.6.3 Generating relations

Relations are generated from connecting predicates, ontology and co-references by implementing the rules explained in Section 5.5.2. Implementation is explained in following sections.

5.6.3.1 Relations based on predicates

All predicates that were used earlier for identification of objects in the graph are stored as a *tuple(Object, predicate)* in the *ObjectMap*. These predicates are given a unique key made of *word-number* and *sentence-number*. We go through the dependency relations to find the corresponding verb-objects for these predicates to implement rule 5.5.2.1.

i. $if\ \exists tuple(X, rel)\ in\ ObjectMap\ and\ Y \in ObjectList$

$and\ \ dobj(rel, Y) \in dependencylist$ then add *relation(X,Y,rel)* in the graph,

ex. In sentence *President counterattacked Iraq.* the *tuple(President, counterattacked)* identified from rule 5.5.1.2 and *Iraq* ∈*ObjectList* from rule 5.5.1.1 and *dobj(counterattacked, Iraq)* ∈ *dependency list* so a new relation is formed *relation(President, Iraq, counterattacked).*

ii. if $\exists tuple(X, rel)\ in\ ObjectMap\ and\ Y \in ObjectList$

$and\ \ iobj(rel, Y) \in dependencylist$ then add *relation (X,Y, rel)* in the graph.

ex. In sentence *Samantha gave Beverly a cake.* the *tuple(Samantha, gave)* identified from Rule 5.5.1.2 and *Beverly* ∈*ObjectList* from Rule 5.5.1.1 and *iobj(gave,cake)* ∈ *dependency list* so a new relation is formed *relation(Samantha, Beverly, gave).*

iii. if $\exists tuple(X, rel)\ in\ ObjectMap\ and\ Y \in ObjectList$

$and\ \ pobj(rel, Y) \in dependencylist$ then Add *relation(X,Y,rel)* in the graph.

ex. In sentence *Hurricane Gilbert swept toward the Dominican Republic.* a new relation is formed *relation(Gilbert, Dominician_Republic, swept_towards).*

iv. If $\exists X, Y \in ObjectList\ and\ \ partmod(rel, X), dobj(rel, Y) \in dependencylist$ then add *relation(X,Y,rel)* in the graph.

ex. In sentence *Strong winds associated with the Gilbert brought coastal flooding. P*artially modifying predicate *associated* derives *relation(wind, Gilbert, associated).*

v. If $\exists X, Y \in ObjectList\ and\ \ poss(X, Y) \in dependencylist$ and then add *relation(X,Y,'possess-0-1')* in the graph.

ex. In sentence *Lucy's dressing room was painted pink.* the dependency relation *poss(Lucy, room)* generates *relation(Lucy, room, possess-0-1)*in the O-O semantic graph. All relations which were not explicitly present in the original text are post fixed with *0-1* key.

vi. if $\exists X, Y \in ObjectList\ and\ nsubj(X, Y) \in dependencylist\ and\ part\ of\ speech(X) = noun, part\ of\ speech(Y) = noun$ then add *relation(X,Y,`is-0-1')* in the graph

ex. In sentence *Pakistan's top diplomat, Bashir Babar, was summoned to India's Foreign Ministry on Sunday.*, Objects *Diplomat* and *Bashir Babbar* gets connected by relation *relation(Bashir Babbar, Diplomat, `is-0-1').*

vi. if two objects are connected by a preposition without any predicate

ex. In sentence *Workers across the European Union are staging a series of protests.* the prepositional relation *pobj_across(Workers, Union)* gets converted to *relation(Workers, European Union, across-0-1)* in the graph. Key *0-1* is added to differentiate it from predicate based relations, since these relations can be further enhanced to semantic labels of *"location", "time"* etc.

5.6.3.2 Ontological relations

We propose in the rule 5.5.2.4 to identify more relations between objects by using an existing ontology, to give deeper semantic understanding to the information presented in the graph. We have used WordNet ontology to find out the synonymous and hypernyms relation between objects.

In the development phase, we have utilized open source *Java WordNet Library*(JWNL) to access WordNet ontology. For every pair of object, first we check for synonymy relation by checking if one of them is present in the *synset* of other object in the WordNet ontology. If this holds true we generate *relation (Object1, Object2, similar-to-0-0).* If synonymous relation is not found then we look for deepest common hypernym of both objects. Y is a hypernym of X, if every X is a kind of Y. We limit the relative depth from common parent node to target object from source object to be 6 nodes. So if the objects are at 6 nodes distance in the ontology and merges at a common parent node then they are connected by *relation (Object1, Object2, kind-of*-0-0) relation.

Ex. In the text *Camera was fixed on a falcon's back. Video was recorded for the bird's movement.* Where co-reference resolver fails to identify *bird* as same reference to *Falcon,* ontological relation *relation(falcon, bird, kind_of-0-0)* helps in connecting the O-O semantic graph objects.

5.6.3.3 Co-reference based relations

For every pair of objects we look into the *wordToMentionMap* and if the objects share same reference they are merged into one object, until they are already connected by a relation that is not ontological relation.

During merging one of the objects becomes the representative object of the two and other object is merged into this object. Deciding the representing object is done either based on strengths of objects or if one of them is an identified named entity. Strength of object is determined by counting the number of properties and operations of object. Since identification of these relations depends on complete information of object, this step is delayed till the end of O-O semantic graph generation. During merging all properties of secondary object are copied to main object, if it does not exist already. Also all relations to secondary object are directed to main object. To reduce information loss secondary object is converted to a property in the main object.

5.6.4 Properties

In this section we explain the extraction of properties of objects and relations.

5.6.4.1 Properties of objects

Rule 5.5.4.1 is implemented by identifying adjectival dependency relations as follows.

i.Add $property(X, Y)$ if $\mathrm{X} \in \mathrm{ObjectList}$ and $\exists\ \mathrm{amod}(\mathrm{X}, \mathrm{Y}) \in dependecyRelationList$

$and\ pos(Y) = adjective$

Ex. In text *The black cat is rolling on the floor.* , *Black* gets added as a property of object *Cat*.

ii. Add $property(X, Y)\ if \exists X, Y \in ObjectList\ and\ nsubj(X, Y) \in dependencylist$

$and\ part\ of\ speech(X) = noun, part\ of\ speech(Y) = adjective$

Ex. In text *The cat was small*. *Small* gets added as property of object *cat*.

iii. Add $property(X, Y)\ if \exists X \in ObjectList\ and\ num(X, Y) \in dependencylist$

$and\ part\ of\ speech(X) = noun, part\ of\ speech(Y) = Cardinal$

Ex. In text *50 computers were distributed to people. 50* becomes a property of object *computer*.

iv. All verbs with no verb-objects become property.

Ex. In text *All businesses were destroyed. Destroyed* becomes a property of object *Business*.

v. Prepositional things either become relation/property.

5.6.4.2 Properties of relations

Clausal complement becomes properties of relation identified from predicates.

$if\ \exists \mathrm{ccomp}(Verb_x, \mathrm{Verb}_y), \mathrm{nsubj}(\mathrm{Verb}_x, \mathrm{A}) \in dependencylist\ and\ (\mathrm{P}, \mathrm{Q}, \mathrm{Verb}_y) \in$ relation *then add property($Verb_Y$, A + $Verb_x$)*

Ex. In the text *John said Maria will resign from IBM*. the identified relation is *relation (Maria, IBM, resign_from)* and added property to the relation is *property (resign, John said).*

These properties help in preserving the context and accuracy of information. Negation and auxiliary information about verbs are prefixed to derived relations from them.

5.6.5 Operation/behaviour

As explained in rule 5.5.3.1 operations of an object are the verbs along with their arguments when they don't connect object with another object. This is simultaneously looked during relation identification from predicates which are explained in Section 5.6.3.1.

if $\exists tuple(X, rel)$ *in* $ObjectMap$ *and* $Y \notin ObjectList$ *and* $dobj(rel, Y) \in dependencylist$ *then add operation rel + prepositional connector (negation Information + auxiliary information + Y) for object X.*

ex. In text *Wind brought coastal flooding, the tuple* The *tuple(Wind, brought) is* identified from Rule 5.5.1.2 and *coastal flooding* is not recognised as object. So the predicate *brought* along with argument *coastal flooding* generates operation *Brought(coastal flooding*) for object *Wind.* In cases of numerical values as verb-object it becomes operation of object. In example text *Big Mac costs $1.14.* operation *costs($1.14)* is generated for object *Big Mac.*

Similarly the cases for indirect object, prepositional objects, and possession dependency relations are handled.

5.6.6 Post processing

After construction of the O-O semantic graph, it is post processed to explore further connections between objects. Objects may be merged in this step. This is different from merging objects of same co-references which is explained in Section 5.6.3.3. In this kind of merging we look into the arguments of operations of objects. If the argument is in possession operation by any other object then both objects are connected by the operation and the argument becomes property of this newly formed relation. This can be seen in text snippet "*Hurricane Gilbert swept toward the Dominican Republic. Residents should closely follow Gilbert's movement."* Here three identified objects are *Gilbert, Residents, and Dominican Republic*. Word m*ovement* is neither a named entity nor it is subject of predicate so it does not belong to objects. In this case the second sentence is broken down into two operations – operation *possess-0-1(movement)* belonging to object *Gilbert* and operation *should_follow(movement)* to object *Resident*. Hence, in post processing we club together

these into a relation *relation (Resident, Gilbert, should_follow: movement).* Here we can see that *movement* has been added as a property of relation *should_follow*.

5.7 Visual representation of O-O semantic graph

All above stated implementation rules from dependency parse to object-oriented semantic graph has been shown on different text snippets. We have run experiments on the DUC text documents for analysis and improvements of O-O semantic graph implementation. Here we have shown a complete text document from the DUC2002 summarisation corpus in Figure 5.4 and its O-O semantic graph generated from our graph generator is shown in Figure 5.5.

Hurricane Gilbert swept toward the Dominican Republic Sunday, and the Civil Defense alerted its heavily populated south coast to prepare for high winds, heavy rains and high seas. The storm was approaching from the southeast with sustained winds of 75 mph gusting to 92 mph. "There is no need for alarm," Civil Defense Director Eugenio Cabral said in a television alert shortly before midnight Saturday. Cabral said residents of the province of Barahona should closely follow Gilbert's movement. An estimated 100,000 people live in the province, including 70,000 in the city of Barahona, about 125 miles west of Santo Domingo. Tropical Storm Gilbert formed in the eastern Caribbean and strengthened into a hurricane Saturday night. The National Hurricane Center in Miami reported its position at 2 a.m. Sunday at latitude 16.1 north, longitude 67.5 west, about 140 miles south of Ponce, Puerto Rico, and 200 miles southeast of Santo Domingo. The National Weather Service in San Juan, Puerto Rico, said Gilbert was moving westward at 15 mph with a "broad area of cloudiness and heavy weather" rotating around the centre of the storm. The weather service issued a flash flood watch for Puerto Rico and the Virgin Islands until at least 6 p.m. Sunday. Strong winds associated with the Gilbert brought coastal flooding, strong southeast winds and up to 12 feet to Puerto Rico's south coast. There were no reports of casualties. San Juan, on the north coast, had heavy rains and gusts Saturday, but they subsided during the night. On Saturday, Hurricane Florence was downgraded to a tropical storm and its remnants pushed inland from the U.S. Gulf Coast. Residents returned home, happy to find little damage from 80 mph winds and sheets of rain. Florence, the sixth named storm of the 1988 Atlantic storm season, was the second hurricane. The first, Debby, reached minimal hurricane strength briefly before hitting the Mexican coast last month.

Figure 5.4: Text document from DUC-2002 summarisation corpus.

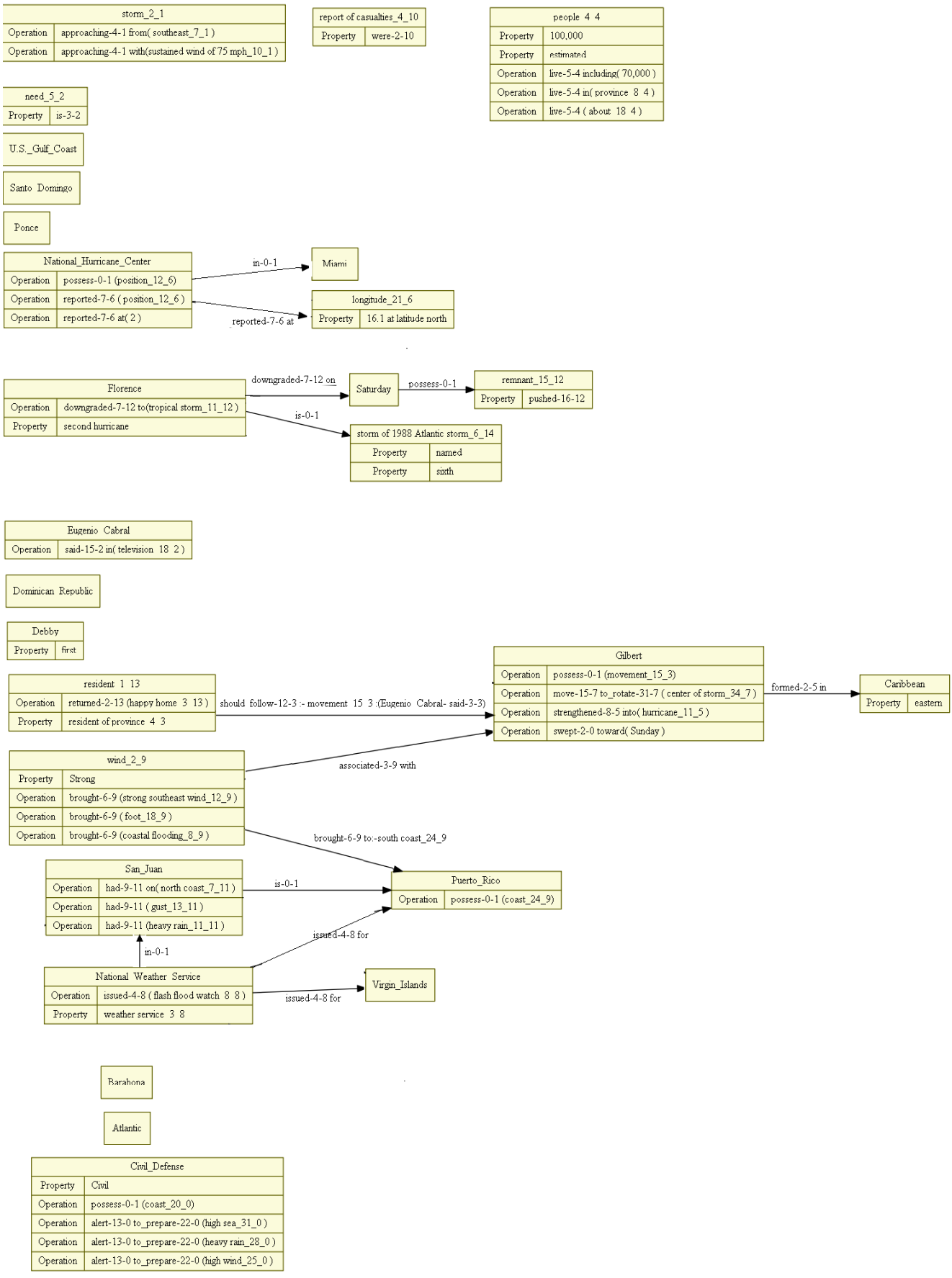

Figure 5.5: Object oriented semantic graph generated for text in Figure 5.4.

This graph is convertible to graph description language DOT and can be used easily by other researchers for further exploration. The graph shown in Figure 5.5 is converted to DOT language as shown in Figure 5.6.

```
[Atlantic]
[Barahona]
[Caribbean]
*eastern
[Civil_Defense]
*Civil
possess-0-1 (coast_20_0)
 alert-13-0 to_prepare-22-0 (high sea_31_0 )
 alert-13-0 to_prepare-22-0 (heavy rain_28_0 )
 alert-13-0 to_prepare-22-0 (high wind_25_0 )
[Debby]
*  first
[Dominican_Republic]
[Eugenio_Cabral]
 said-15-2 in( television_18_2 )
[Florence]
 downgraded-7-12 to(tropical storm_11_12 )
*second hurricane
[Gilbert]
possess-0-1 (movement_15_3)
 move-15-7 to_rotate-31-7 ( center of storm_34_7 )
 strengthened-8-5 into( hurricane_11_5 )
 swept-2-0 toward( Sunday )
[Miami]
[Ponce]
[Puerto_Rico]
possess-0-1 (coast_24_9)
[San_Juan]
 had-9-11 on( north coast_7_11 )
 had-9-11 ( gust_13_11 )
 had-9-11 (heavy rain_11_11 )
[Santo_Domingo]
[Saturday]
[U.S._Gulf_Coast]
[Virgin_Islands]
[longitude_21_6]
*16.1 at latitude north
[need_5_2]
* is-3-2
[people_4_4]
*100,000
*estimated
 live-5-4 including( 70,000 )
 live-5-4 in( province_8_4 )
 live-5-4 ( about_18_4 )
```

Figure 5.6 (i): Textual representation of object-oriented semantic graph in graph description language DOT.

```
[remnant_15_12]
* pushed-16-12
[report of casualties_4_10]
* were-2-10
 [U.S._Gulf_Coast]
[Virgin_Islands]
[longitude_21_6]
*16.1 at latitude north
[need_5_2]
* is-3-2
[people_4_4]
*100,000
*estimated
 live-5-4 including( 70,000 )
 live-5-4 in( province_8_4 )
 live-5-4 ( about_18_4 )
[remnant_15_12]
* pushed-16-12
[report of casualties_4_10]
* were-2-10
[storm of 1988 Atlantic storm_6_14]
*named
*sixth
[storm_2_1]
 approaching-4-1 from( southeast_7_1 )
 approaching-4-1 with(sustained wind of 75 mph_10_1 )
[wind_2_9]
*Strong
 brought-6-9 (strong southeast wind_12_9 )
 brought-6-9 ( foot_18_9 )
 brought-6-9 (coastal flooding_8_9 )
[National_Hurricane_Center]
possess-0-1 (position_12_6)
 reported-7-6 ( position_12_6 )
 reported-7-6 at( 2 )
[National_Weather_Service]
 issued-4-8 ( flash flood watch_8_8 )
*weather service_3_8
[resident_1_13]
 returned-2-13 (happy home_3_13 )
*resident of province_4_3
[Florence] 1--* [Saturday] <  downgraded-7-12 on>
[Gilbert] 1--* [Caribbean] <  formed-2-5 in>
[National_Hurricane_Center] 1--* [longitude_21_6] <  reported-7-6 at>
[San_Juan] 1--* [Puerto_Rico] < is-0-1>
[Saturday] 1--* [remnant_15_12] <possess-0-1>
[Florence] 1--* [storm of 1988 Atlantic storm_6_14] < is-0-1>
[National_Weather_Service] 1--* [Virgin_Islands] <  issued-4-8 for>
[National_Weather_Service] 1--* [Puerto_Rico] <  issued-4-8 for>
[wind_2_9] 1--* [Gilbert] <  associated-3-9 with>
[wind_2_9] 1--* [Puerto_Rico] <  brought-6-9 to:-south coast_24_9 >
[resident_1_13] 1--* [Gilbert] <should  follow-12-3 :- movement_15_3 :(Eugenio_Cabral- said-3-3)>
[National_Hurricane_Center] 1--* [Miami] <in-0-1>
[National_Weather_Service] 1--* [San_Juan] <in-0-1>
```

Figure 5.6(ii): Textual representation of object-oriented semantic graph in graph description language DOT.

5.8 Evaluation

It is not possible to conduct a comparative evaluation because this is first work on object oriented semantic graph generation from unrestrained text. Here we have manually evaluated the graph by getting recall of correctly identified objects and relations. We went through each sentence of the text shown in Fig. 5.4 and manually checked for possible objects and relations. A total of 37 objects can be manually identified from the text. 34 objects are correctly identified from our implementation shown in Figure 5.6(i) and 5.6(ii) and the missing objects are *southeast, Mexican coast* and *civil defence director.* It gives recall rate of 91.9% in identifying objects from the text.

We manually looked for relations/operations generated from predicates in the text and compared it with system generated relations/operations from predicates in the O-O semantic graph. Manually we have identified 54 relations/operations from predicates. In our implementation, we got a total of 50 relations/operations from predicates. 4 missing relations are (i) *Debby hitting Mexican coast*, (ii) *National Hurricane Center reported position about 140 miles south of ponce* (iii) *Gilbert moving westward at 15mph* and *(iv) heavy rains and gusts subsided during night*. Recall rate in identifying relations/operations from predicate is 92.6%.

Similar evluation is done on another document shown in fig 5.7. Corresponding O-O semantic graph is shown in fig. 5.8 , wheras the textual represenation of graph is shown in fig 5.9(i) and (ii). Manual analysis gave 35 objects and the graph had 41 objects. Which shows the recall is 100% but the precision is 85.3% as some non-object text also gets converted to object which are *i. Olympic_Saddledome, ii. 50_percent, iii. repeatSingles, iv. Carmen v. performance and vi. Debi Thomas Bronze.*

Debi Thomas' dream of Olympic gold turned into disappointment Saturday as East Germany's Katarina Witt won her second straight Olympic championship and Canadian Elizabeth Manley took home the silver before a crowd of cheering countrymen. ``It's over. Back to school,'' said Thomas, who won the bronze medal despite three faulty landings. ``I'm not going to make any excuses. I was really skating well this week. It wasn't supposed to happen, I guess. But I tried.''

While the top two skaters in the world staged a shootout to music from Bizet's ``Carmen,'' Manley was so sensational in the freestyle that she finished first with seven judges. Combined with a fourth in the compulsory figures and a third-place finish in the short program earlier in the week, the performance put Manley in second place.

Witt, a three-time world champion from East Germany, became the first repeat singles champion since Dick Button took Olympic gold in 1948 and '52. Sonja Henie of Norway was the only woman to do it before Witt, winning in 1928, 1932 and 1936. Thomas, of San Jose, Calif., the first black to win a U.S. figure skating crown and the 1986 world champion, skated poorly Saturday after doing well earlier in the Games. By contrast, Manley had the sellout crowd at the Olympic

Saddledome enraptured. They cheered, hooted and stamped their feet when she finished hitting every element of her program. Jill Trenary of Minnetonka, Minn., finished fourth. She was fifth heading into the long program, worth 50 percent of the overall score. Thomas' bronze was the third figure skating medal here for the United States. Brian Boitano won the men's crown, and a bronze in pairs went to Jill Watson and Peter Oppegard.

In addition to the three figure skating medals, the U.S. team had three speed-skating medals: one each gold, silver and bronze. Speed skater Bonnie Blair, America's only double medalist, tried again Saturday in the 1,500 meters but finished fourth, well off the pace. She won the gold in the 500 and the bronze in the 1,000 meters.

As the Olympics winded up its next-to-last day, the Soviet Union had 27 medals, including 11 golds, while East Germany in second place had 22, including nine golds.

Figure 5.7 Text document from DUC-2002 summarisation corpus

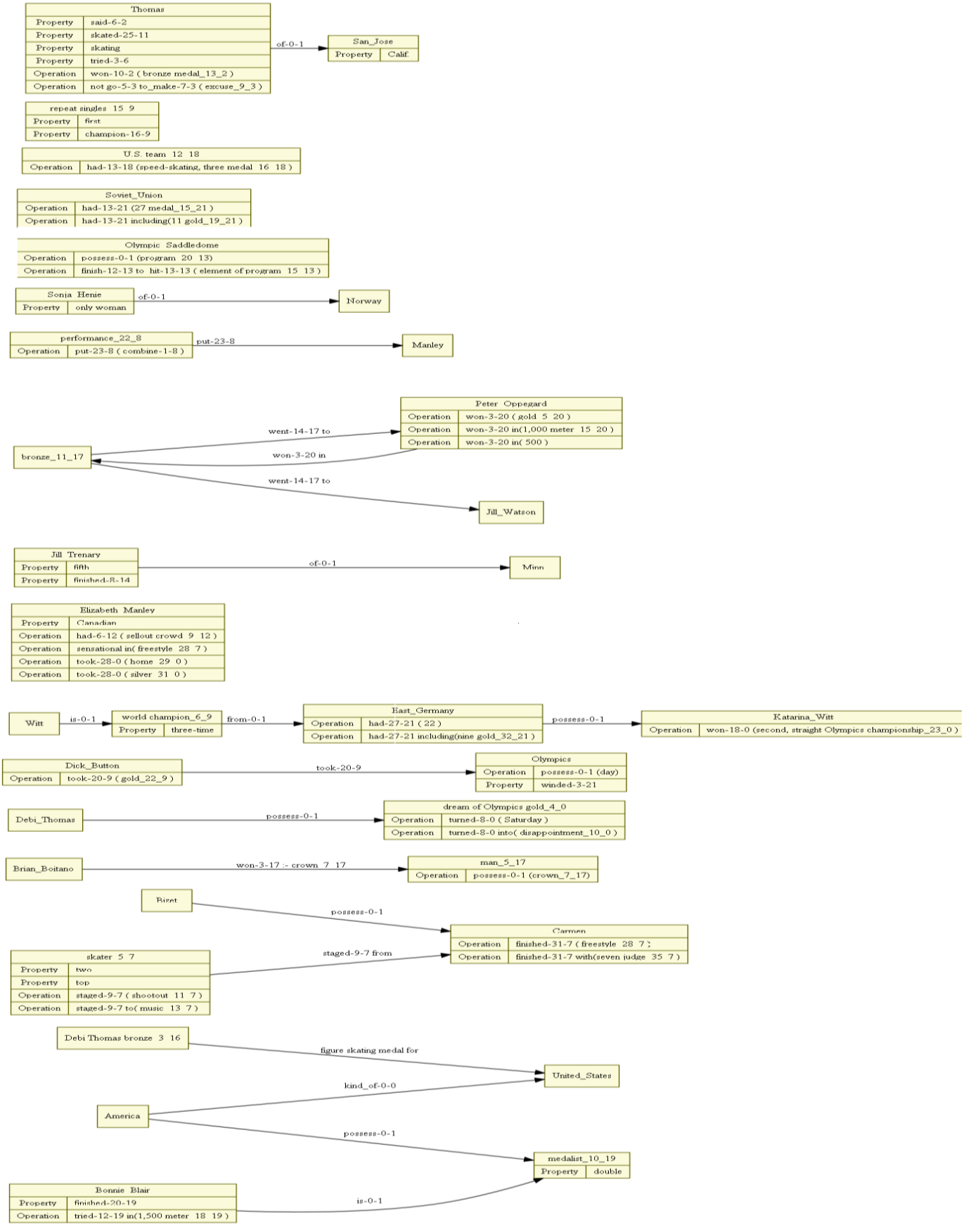

Figure 5.8 Object oriented semantic graph generated for text in Figure 5.7.

```
[America]
[Bizet]
[Bonnie_Blair]
* finished-20-19
 tried-12-19 in(1,500 meter_18_19 )
[Brian_Boitano]
[Carmen]
 finished-31-7 ( freestyle_28_7 )
 finished-31-7 with(seven judge_35_7 )
[Debi Thomas bronze_3_16]
[Debi_Thomas]
[Dick_Button]
 took-20-9 ( gold_22_9 )
[East_Germany]
 had-27-21 ( 22 )
 had-27-21 including(nine gold_32_21 )
[Elizabeth_Manley]
*Canadian
 had-6-12 ( sellout crowd_9_12 )
 sensational in( freestyle_28_7 )
 took-28-0 ( home_29_0 )
 took-28-0 ( silver_31_0 )
[Jill_Trenary]
* fifth
* finished-8-14
[Jill_Watson]
[Katarina_Witt]
 won-18-0 (second, straight Olympics championship_23_0 )
[Manley]
[Minnetonka]
[Norway]
[Olympic_Saddledome]
possess-0-1 (program_20_13)
 finish-12-13 to_hit-13-13 ( element of program_15_13 )
[Olympics]
possess-0-1 (day)
* winded-3-21
[Peter_Oppegard]
 won-3-20 ( gold_5_20 )
 won-3-20 in(1,000 meter_15_20 )
 won-3-20 in( 500 )
[Sonja_Henie]
*only woman
[Soviet_Union]
 had-13-21 (27 medal_15_21 )
 had-13-21 including(11 gold_19_21 )
[U.S.]
[U.S. figure_16_11]
* skating
[U.S. team_12_18]
 had-13-18 (speed-skating, three medal_16_18 )
[United_States]
[Witt]
[bronze_11_17]
```

Figure 5.9(i): Textual representation of object-oriented semantic graph in graph description language DOT.

```
[dream of Olympics gold_4_0]
 turned-8-0 ( Saturday )
 turned-8-0 into( disappointment_10_0 )
[man_5_17]
possess-0-1 (crown_7_17)
[medalist_10_19]
*double
[performance_22_8]
 put-23-8 ( combine-1-8 )
[repeat singles_15_9]
*first
* champion-16-9
[skater_5_7]
*two
*top
 staged-9-7 ( shootout_11_7 )
 staged-9-7 to( music_13_7 )
[world champion_6_9]
*three-time
[San_Jose]
*Calif.
[Thomas]
* said-6-2
* skated-25-11
* skating
* tried-3-6
 won-10-2 ( bronze medal_13_2 )
not go-5-3 to_make-7-3 ( excuse_9_3 )
[America] 1--* [medalist_10_19] <possess-0-1>
[Bizet] 1--* [Carmen] <possess-0-1>
[Brian_Boitano] 1--* [man_5_17] <  won-3-17 :- crown_7_17 >
[Debi Thomas bronze_3_16] 1--* [United_States] <  figure skating medal for>
[Debi_Thomas] 1--* [dream of Olympics gold_4_0] <possess-0-1>
[Dick_Button] 1--* [Olympics] <  took-20-9 >
[East_Germany] 1--* [Katarina_Witt] <possess-0-1>
[Peter_Oppegard] 1--* [bronze_11_17] <  won-3-20 in>
[bronze_11_17] 1--* [Jill_Watson] <  went-14-17 to>
[bronze_11_17] 1--* [Peter_Oppegard] <  went-14-17 to>
[Bonnie_Blair] 1--* [medalist_10_19] < is-0-1>
[performance_22_8] 1--* [Manley] <  put-23-8 >
[skater_5_7] 1--* [Carmen] <  staged-9-7 from>
[Witt] 1--* [world champion_6_9] < is-0-1>
[Jill_Trenary] 1--* [Minn.] <of-0-1>
[Sonja_Henie] 1--* [Norway] <of-0-1>
[Thomas] 1--* [San_Jose] <of-0-1>
[world champion_6_9] 1--* [East_Germany] <from-0-1>
[America] 1--* [United_States] <kind_of-0-0>
```

Figure 5.9(ii): Textual representation of object-oriented semantic graph in graph description language DOT.

In next Chapter we will show the evaluation of object oriented semantic graph in abstractive summarisation.

5.9 Conclusion

In this Chapter we have looked into the different ways of modelling natural language text and various semantic graph based representations used in natural language processing. We have analysed the reasons why modelling textual content is difficult due to complexities in sentence formation. Sentences can vary from a simple subject and predicate to a complex sentence of many clauses. Transforming textual content to a graphical representation which preserves the semantics and modularises information into clear, non-ambiguous subparts is an ambitious goal. We have proposed our solution in the form of object-oriented semantic graph and have provided detailed rules and their implementations to generate the graph. This graph can then be utilised for various NLP task. In next Chapter we will describe the use of this proposed graph in text summarisation task.

Chapter 6

Abstractive Summary Generation from O-O Semantic Graph

In the previous Chapter we have presented the design and development of object-oriented semantic graph, a representation of text document from object-oriented analysis and design principles. Object-oriented semantic graphs can be generated from text according to a ruleset that has been designed based on linguistic knowledge and works on dependency relations of the word units in the sentences. The linguistic knowledge to shape these rules was acquired after understanding past work on modelling text data and the grammatical connection in sentences. The ruleset will require further refinement based on more scenario analysis before reaching a mature stage, but nevertheless it is enough to demonstrate the feasibility of using object-oriented semantic graph for text summarisation.

Summarisation is a natural language process to convert long text documents such as online news articles, scientific papers, and patent text documents into a short form for various user purposes such as reading from a small handheld device or getting a quick overview of the document. This is the main objective of our research work. Chapter 2 provides a detailed literature review on text summarisation. As we can see this process has been researched under the heading “automatic text summarisation” for many years and is still an active research area. Text summarisation can be of different types according to the summary generation strategy: extractive or abstractive. It also varies according to information content it is applied to: single document summarisation, multiple document summarisation and opinion/review summarisation. Among these summarisation categories extractive

summarisation of different types of information content is better researched than abstractive summarisation due to less complexity involved in this method [9,86]. In this approach sentences are selected from the original document for inclusion in summary without any modification based on the ranking of sentences by various heuristic i.e. graph based, sentence similarity, position. On the other hand the preferable and closest to the human way of summary generation is abstractive summarisation, where new sentences should ideally be formed based on an understanding of the structure and the content of the document by taking only the important parts of the document. Abstractive summary sentences can be completely different from original sentences, or can be similar to them depending on the abstraction scheme. A critical literature review of text summarisation revealed the limits of extractive summarisation, which are dangling references, lack of coherence between summary sentences. The literature review also found abstractive or hybrid text summarisation desirable as it is based on an understanding of text and generates summaries that are more coherent than extractive summaries when evaluated against human written summaries [17,86].

6.1 Abstractive summarisation

In most of the existing work on abstractive summarisation [123,125,133,155] summary generation is based on sentence compression and has been solved as an optimization problem using integer linear programming. Sentence compression is focussed on sentence reduction not on document reduction and has been generally tackled by supervised methods trained on parallel corpuses of sentences along with their compressed counterparts [123,155]. Supervised sentence compression methods are not portable to other domains where training data is not available due to scarcity of parallel corpuses, although there is ongoing research work focussed on automatically creating parallel corpuses by web crawling for news articles

about same events [212]. Some unsupervised methods [213,214] perform sentence compression by using hand-crafted or learnt rules of parse tree reduction or words deletion. Overall primarily it has been based on syntactic and statistical features to delete words from sentences while often ignoring context information that is crucial to form a coherent summary. Some approaches have tried to include contextual information from discourse for overall document compression [121,215].

Abstractive approaches for guided summarisation has concentrated on compression techniques by using graph cuts for joint compression and summary generation[125] and for pipeline compression-extraction approaches, where some concept words are identified in-hand to preserve during compression[123]. Yet the performances of joint or pipelined extraction-compression methods are delimited by the drawbacks of extractive summarisation and lack of parallel data[11,212].

Deeper semantic approach to abstractive guided summarisation[216] works by using hand written information extraction rules. These rules are based on syntactic analysis of sentences to fill the information about different aspects of many categories in that document. In multi-document summarisation abstractive approaches have used sentence fusion[45], semantic role labelling [133] to generate summary from many similar documents about same event or topic. For generic single document summarisation the abstractive approaches have been less researched compared to extractive approaches. It has been formulated as optimization of sentence compression-extraction pipeline and has been solved by integer linear programming [217]. One of the single document abstractive summarisation approaches, *compendium* [14], applies compression and fusion on word graphs generated from extractive summaries to produce abstractive summaries.

Abstractive summarisation requires interpretation of source text and its representation to generate summaries from it. Very few approaches generate summary from representation of

original document. Summary generation directly from reduced vector space representation after *LSA* analysis is a step towards it [143], where complete sentences are reconstructed from the salience word in LSA model following a noisy channel framework. A recent summarisation approach has been shown by utilising newly developed deeper semantic representation *Abstract Meaning Representation* (AMR)[74]. It is a supervised approach focused on graph reduction from AMR representation of source text and text generation step is left for future study. Our approach works from representation to text generation and is an unsupervised approach primarily focussed on single document summarisation. We have implemented all three steps (interpretation, graph generation and text generation) and unlike other approaches, we try to remove reference ambiguities by utilising co-reference resolution. It is unsupervised approach so it can be applied to any corpus that may not have sufficient training examples for other abstractive summarisation approaches such as sentence compression.

6.2 Proposed approach

In our abstractive summarisation approach we aim to explore the content of document interactively at various levels of abstraction, from a list of ranked topics to details of the important topics. To achieve this we first interpret and represent the complete document from an object oriented paradigm and then generates summary from this representation. This kind of summarisation does not face impediment like scarcity of training data and utilises cross-sentence discourse information as it is based on a document level representation instead of independent sentence level representation. Our proposed approach conforms to the research direction for summarisation: interpretation of text, representation of text as a connected whole unit and generation of summary from the representation. This approach is a new

contribution in abstractive summarisation research, as it is first methodology to perform summary generation from object-oriented transformation of text content.. The closest other work is summarisation based on AMR graphs. AMR graphs utilised for summarisation are generated from a trained parser on manually annotated sentences, whereas the O-O semantic graph generation is unsupervised approach based on linguistic and semantic knowledge..

We use object-oriented semantic graph of text document as the intermediate representation to perform abstractive text summarisation on it. All the information required to generate the summary is stored in the semantic graph and the original document is not any more required after the graph has been constructed. It shows the usefulness of graph as a substitute to the original document. Details of O-O semantic graph are provided in Section 5.5 of Chapter 5. The basic strategy that we follow to generate a summary is to identify important paths in the O-O semantic graph. A path $Path_{i,k}$ in the O-O semantic graph *(G=(V,E))* is defined in following equation.

$$path_{i,k} = \{Object_{i,} relation_{i,i+1}, Object_{i+1}, relation_{i+1,i+2}, \ldots\ldots\ldots\ldots relation_{k-1,k}, Object_k\} \quad (6.1)$$

In Equation (6.1) $Object_i \in \{V\}$ and $relation_{i,i+1} \in \{E\}$. The property of a path is that objects cannot be revisited. Since O-O semantic graph is a multigraph, there can be different paths from $Object_i$ to $Object_k$ between same sequences of objects by visiting different relations between them. So we want to explore all possible paths and rank them based on their significance in a network topology and the significance of semantic information they contain. Importance of each path is determined by the importance of the object nodes and relations connecting them, which has been explained in the next subsections.

6.2.1 Ranking of paths

Various subgraph finding approaches formulate it either as a graph division problem (e.g., max flow min cut problem), or as a graph clustering problem. Clustering could be a good solution for the dense graphs by finding centroids from the graph. But, since the O-O semantic graphs are generally not dense, clustering cannot be a good solution. We decided to use path ranking according to mixed heuristics from graph theory and knowledge of semantics presented through the graph. We first estimate the importance of objects of the graph based on its interaction with other objects in the graph using *PageRank* method to rank the objects based on the numbers of incoming and outgoing relations it has with other objects.

PageRank analysis of graph has been explained in Section 4.4.1 of Chapter 4. *PageRank* score of a node is calculated from the *PageRank* scores of all incoming nodes to it by combining the equally distributed scores of all incoming nodes among their outgoing nodes. *PageRank* score, PR_{node_i}, of any node, $node_i$, is computed from Equation (6.2).

$$PR_{node_i} = (1-d) + d \times \sum_{node_j \in In(node_i)} \frac{PR(node_j)}{\deg(node_j)} \tag{6.2}$$

where d is the dampening factor to accommodate random jumps which is generally set to 0.85[218]. $In(node_i)$ is the set of nodes with incoming links to $node_i$ and $deg(node_j)$ is the degree of outgoing links from $node_j$.

Here we want to bring back to the reader's attention that in O-O semantic graph a relation is verb/ontology/prepositional relation. Although the relation has been projected as a directed edge, it can be considered bi-directed during ranking because in natural language a relation can be made in active voice or passive voice and thus we assume A→R→B ⇔ B→$R_{passive}$→A. O-O semantic graph is converted to bi-directional graph to calculate *PageRank* score of objects. Final importance scores of objects are calculated by combining

the semantic importance of that object with the *PageRank* metric. Semantic importance of an object is computed from the frequency count of its properties and operations. Every object has certain number of properties and operations that describes the way it has been given importance in the document. More number of properties and operations indicates it was described in detail in the original text and has much priority than the lesser described objects. We also call it *strength of the object* and it is calculated by Equation (6.3).

$$Strength(Obj_i) = Count(properties \in Obj_i) + Count(operations \in Obj_i) \quad (6.3)$$

We combine this strength of the object with the *PageRank* score *PageRank(Obj$_i$)* to get the final importance score of the object, as explained in Equation (6.4).

$$Rank(Obj_i) = PageRank(Obj_i) + \frac{Strength(Obj_i)}{\underset{i}{Max}\, Strength(Obj_i)} \quad (6.4)$$

A path is made up of objects and relations. So the importance of both parts contributes to the final approximation of path importance. Importance of a relation *relation$_p$* is computed by counting its occurrence in the final graph. We check for the occurrence in all relations and operation.

$$popularity(relation_p) = \log(Frequency(relation_p)) + \log(Frequency(relation \approx relation_p)) \quad (6.5)$$

First frequency is count of same relation in the graph and second frequency is count of relations which are similar to this relation. Logarithm of frequency counts will save from giving more priority to relations with many arguments. Relations (verbs) with many arguments (indirect object, direct object, prep object) may have more frequency count.

Importance score of a path is computed from the Equation (6.6).

$$pathScore_{i,j} = PageRank(Node_i) + popularity(relation_{i,i+1}) + PageRank(Node_{i+1}) + 0.1^{m+1}(pathScore_{i+1,i+2}) \quad (6.6)$$

Since we aim to reduce the graph to an optimised subgraph, we penalise longer paths and look for shorter paths between all objects. We explore shortest path between all of the objects in a directed graph and rank the paths by Equation (6.6).

Similar to paths the operations of objects are integrated in the summary according to their presence in the already constructed summary or according to the raking of the objects. Next Section describes the ranking of operations.

6.2.2 Ranking of operations

Operations as described earlier in Section 5.5.3 of Chapter 5 are those verbs whose only one argument is an object. Remaining arguments of this verb are not objects in the graph. This gives different ways to compute importance of the operations. We have tested two different ways and have analysed the effect on generated summaries.

1. Considering operations similar to paths with only one object, we follow Equation (6.6) for ranking of operations and paths both. In case of operations the score of second object is considered nil.

2. We connect the importance of operations to already ranked paths. If the main verb of operations has ranked high in the path scoring, then same rank is given to this operation.

These rankings play a substantial role in generating summary from the graph. We will see in the results Section the impact of these different rankings on summary generation.

6.2.3 Ranking of properties

A lot of information also goes in the properties of objects. O-O semantic graph generation scheme makes sure that no property gets duplicated. Ranking of properties has not been

researched in our current work. But it can be done using conceptual similarity to objects. Frequency count cannot be applied as no property is duplicated within the same object.

6.3 Constructing summary sentences

In earlier sections we described the methodology to rank the different units of graph by combining semantics and graph metrics. In this Section we provide the summary generation strategy from the O-O semantic graph.

6.3.1 Sentence formation from ranked paths

The final step of our abstractive summarisation approach is realising summary sentences by utilizing the ranking information and natural language generation rules. We list all paths and divide them into smallest units- *path triples(object-relation-object),* and *path-pair(object-operation)* because we want to construct the natural language sentences from these paths. Here an interesting observation is that many paths will have common path triples, because many paths may have been routed through same subset of objects and relations. To avoid redundant information to be added in the summary we remove duplicate path-triples.

There are various restrictions for summary generation. First restriction is that the newly formed sentence should be grammatically correct. In this regard we have preserved prepositional connectors($prep_k$) and other grammatical connectors in the generated senetneces as shown in eq. 6.7. Thus the generated sentences will not have any advantage over extractive summaries in evauations , since it also includes similar connector words. Second restriction is that sentences should be short and non-redundant. Easiest way to generate sentences would have been converting each path-triple and each pair *(object-operation)* to a sentence. This

kind of sentence generation guarantees to be grammatical but it contradicts the main purpose of shortening the original text to smaller text. Because it will result in some objects being repeatedly referred to in short sentences. For example directly generated sentences from a smaller part of O-O semantic graph shown in Figure 5.5 of Chapter 5 are:

1. National_Weather_Service issued-4-8 for Virgin_Islands

2. National_Weather_Service issued-4-8 for Puerto_Rico

In both sentences the object National_Weather_Service is used, which is clearly repetitive. We resolved this issue by the fusion of newly generated sentences if they share a common relation.This is done to make summary readable. As described earlier every path is divided into smallest units path-triples P_i={$Object_{i1}$,$relation_i$-w_i-s_i,$Object_{i2}$} where w_i , s_i are word number and sentence number of the word that becomes the relation. Word number and sentence numbers are attached to the relations to differentiate relations made of same word in different sentences. Any two paths triples P_j, P_k gets merged into one path-triple P_{merged} according to Equation (6.7).

$$\forall P_j, P_k \{P_{merged} = Object_{j1}, relation_j - w_j - s_j, Object_{j2}, prep_k Object_{k2} \mid Object_{j1} = Object_{k1}, relation_j - w_j - s_j = relation_k - w_k - s_k\} \quad (6.7)$$

The condition to merge triples is that both triples should have similar first object and similar connecting relation. Prepositional connector ($prep_k$) is taken from properties of relation ($relation_k$) to combine the second object of triples. Presence of prepositions makes the sentence grammatically correct. The final sentence after merging the path triples 1 and 2 according to Equation (6.6) is shown below.

National Weather service issued for Virgin Islands, for Puerto Rico.

We have decided to fuse only same relations of similar subjects. Otherwise combining dissimilar relations with similar subjects may lead to creation of very long sentences and

many lower ranking path-triples may reach to higher position which is not advisable when generating short summaries.

As explained in Section 5.6.3 of Chapter 5 three types of relations can be observed in O-O semantic graph: relations from original text based on predicates, relations from ontology and relations that were self-generated *(possess, is*). While generating sentence from a long path, we ignore all the triples which consists an ontological relation (i.e. *Falcon-is_a-Bird*). The reason is that ontological relations are for computation purpose but otherwise these relations are implicitly understandable by humans. So although their ranking still contributes to the overall score of path these relations are not added in the generated summary sentences. Self-generated relations are merged with other triples to make a grammatical sentence later. Earlier in the graph we convert "*possession*" relation between two nouns to a relation of possess in the O-O semantic graph, if both nouns are identified as objects in the graph. To convert this back to summary, we identify if any possess relation has been added to summary. We convert the cases *A→possess→B; B→relation→C* to *A's B→relation→C* for the first occurrence of B according to Equation (6.8). It is to be noted that self-generated relation *possess* in the O-O semantic graph is differentiated from other relation generated from predicate *possess* in the text by adding unique identifier.

$$\forall relation_i, if\ relation_i \in \{possess, is, prep_{relations}\} then\ merge\ relation_i \text{ with first occurence of objects in highest ranking paths} \quad (6.8)$$

Example: *John possess (0-1) car. Car took (2-1) Mark to hospital.*

Here after merging the sentence becomes

John's car took Mark to hospital.

Similarly *"apposition"* relation becomes *"is"* relation in the O-O semantic graph, which is shown below.

Obama is(0-1) President of America.

Obama went to Paris.

After merging the above two sentences according to Equation (6.8) the following new sentence is generated.

Obama, President of America, went to Paris.

Similarly we merge prepositional relations as shown below.

IBM in America.

IBM bought Zynto System.

Resulting sentence is:

IBM in America bought Zynto Systems.

This kind of merging may not be perfectly grammatical, but as we see in experimental Section, it is reasonable grammatical and readable.

6.3.2 Sentence formation from operations of object

Having discussed constructing sentences from ranked paths, we now concentrate on operations of objects in the O-O semantic graph to generate further summary sentences. We consider two methods for inclusion of operations into summary. In the first case paths and operations are considered of equal importance and operations are ranked similarly as paths. Operations and paths are sequenced according to their scores and added to summary following the strategy described in Section 6.3.1 for sentence generation from paths. This setup is described as *Operations ranked As paths.*

In the second case operation are considered less informative than paths and added into summary after summary sentences has been generated from paths. In this case operations can be merged to the already constructed summary sentences if their verb and object match to any previously constructed sentence. In this case operations are added according to the ranking of objects. This case is described as *Operations ranked After paths.*

6.3.3 Applying edit distance as redundancy removal system

After we have generated summary sentences, we perform post processing to remove lower ranked sentences which are similar to sentences ranked higher. Sentence redundancy is its similarity to the remaining sentences, so we use edit distance between the sentences as a measure of redundancy between them. The lesser the edit distance the more similar are the sentences. Removing closely similar sentences promotes information diversity in the summary. We have varied the edit distance threshold from 1 to 5 for sentence removal and have observed the effect of it in summarisation results. Figure 6.1 shows a small simplified news article taken from a website http://www.newsinlevels.com/.

> Serbian construction workers were digging when they found something shocking. It was an unexploded bomb from World War II. The bomb was buried six meters underground. Experts say that the one-ton bomb was made in Germany. It contains 620 kilograms of explosives. Local residents were evacuated from the area and the bomb was transferred to a military base where the bomb will be destroyed safely.

Figure 6.1: Simplified news taken from *NewsInLevel* website.

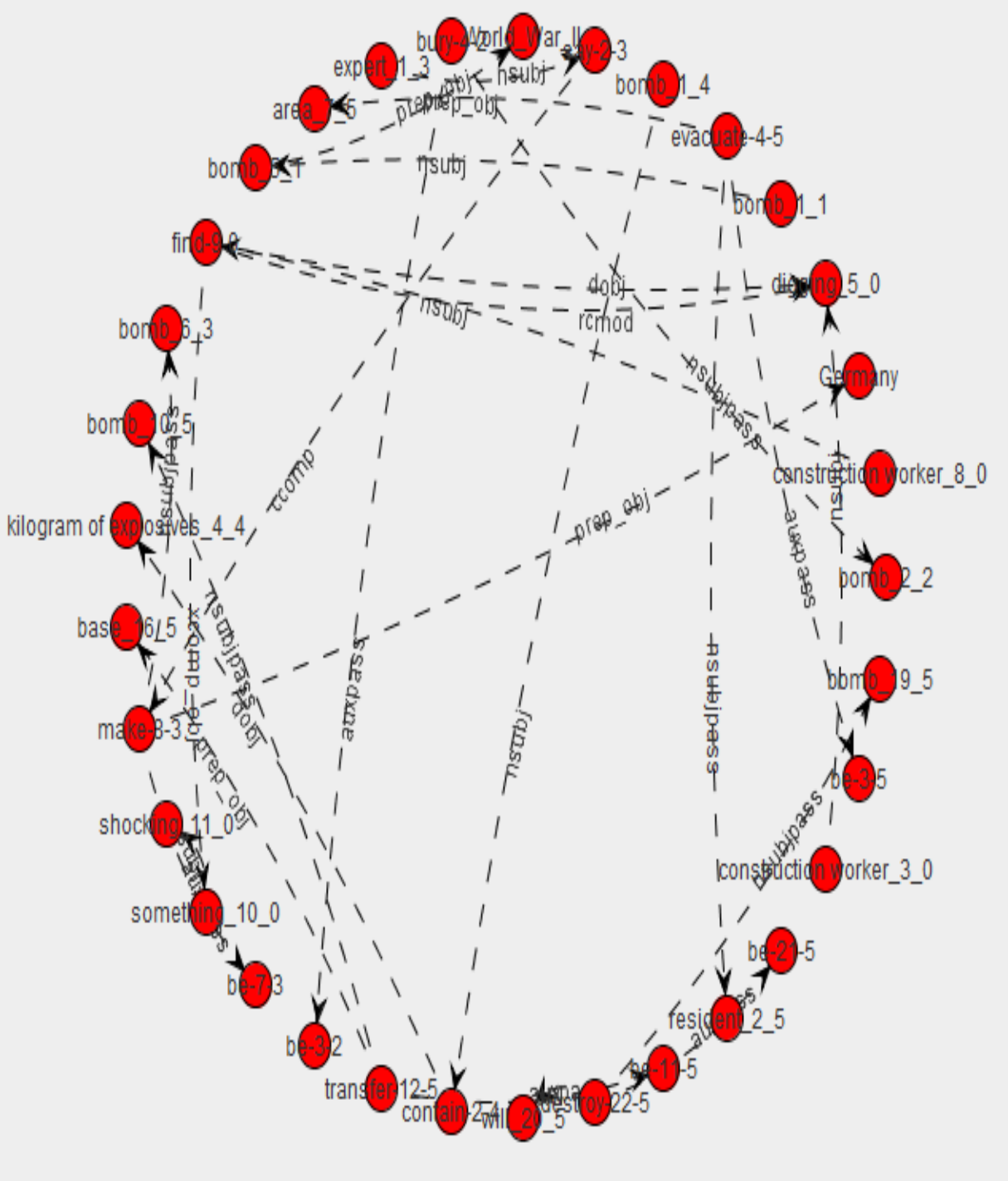

Figure 6.2: Intermediate graph.

Figure 6.2 shows the intermediate graph generated from JUNG library, on which we work and derive our O-O semantic graph by applying all rules explained earlier in Chapter 5. The O-O semantic graph is shown in Figure 6.5 and the abstractive summary generated from it is shown in Figure 6.4.

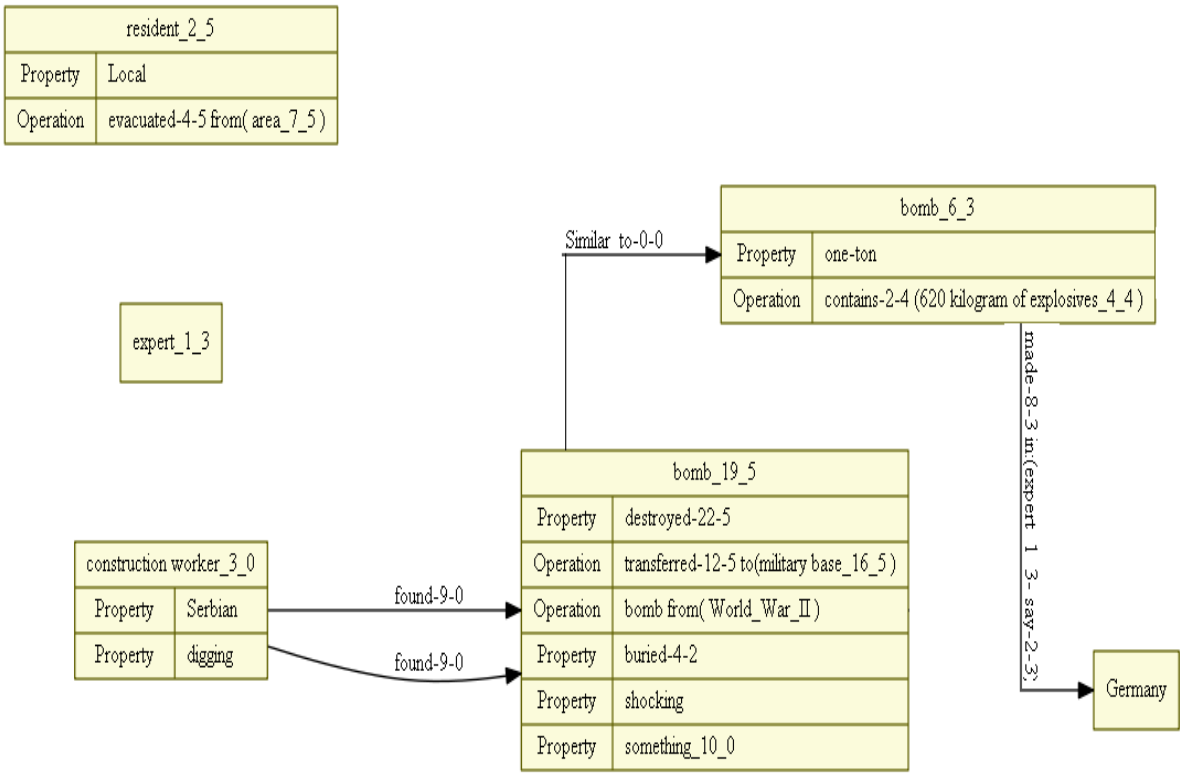

Figure 6.3: O-O semantic graph.

Serbian, digging construction worker found destroyed, buried, shocking, something one ton bomb.

bomb made in Germany expert say.

bomb from World War II .

Figure 6.4: Generated summary.

After that we have applied abstractive summarisation to a bigger document taken from DUC2002 corpus. This document is shown in Figure 5.4 of Chapter 5. Its O-O semantic graph is shown in Figure 5.5 of Chapter 5. The summary of this document by our system is shown in Figure 6.5. Comparison of this summary with the human authored summaries for

the given document in DUC corpus shows that sentences a, b, c, d, f, g and k had similar content as human selected important information while sentences e, h, i, and j were not close to any human selected information content. We can see that some sentences are not grammatically correct. Often the preposition and verb-arguments are not correctly sequenced after the verb position.

a). National Weather Service in San Juan issued flash flood watch for Virgin Islands, Puerto Rico.

b). National Hurricane Centre in Miami reported at 2, it's position longitude.

c). Florence , storm of 1988 Atlantic storm downgraded on Saturday to tropical storm.

d). Strong wind brought coastal flooding, strong southeast wind to south Puerto Rico's coast.

e). resident should follow Gilbert's movement Eugenio Cabral said

f). Gilbert formed in eastern Caribbean

g). Civil Defense alert to prepare for wind, heavy rain and high sea.

h). estimated 100,000, people live in province including 70,000(*missing information*) about

i). Gilbert swept toward Sunday *(wrong generation)*

j). Gilbert move to rotate center of storm.

k). Gilbert strengthened into hurricane.

Figure 6.5: Generated summary.

We have manually corrected the grammar of these generated sentences and shown the manually corrected summary in Figure 6.6. It shows that if improvements are made at natural language generation step, the generated summaries from O-O semantic graph can be more human understandable.

National Weather Service in San Juan issued for Virgin Islands for Puerto Rico flash flood watch

National Hurricane Center in Miami reported at 2 National Hurricane Center's position longitude

Florence , storm of 1988 Atlantic storm downgraded on Saturday to tropical storm

Strong wind brought coastal flooding strong southeast wind to south Puerto Rico's coast

resident should follow Gilbert's movement Eugenio Cabral said

Gilbert formed in eastern Caribbean

Civil Defense alert wind to prepare heavy rain high sea

100,000, estimated people live including 70,000 in province about

Gilbert swept toward Sunday

Gilbert move to rotate center of storm

Gilbert strengthened into hurricane

Figure 6.6: manually corrected summary 1.

A further variation of O-O semantic graph is considered by including frequent verb-objects as objects in the graph. We have also varied the inclusion of adjectives and adverbs in the summary and have looked at the effects on summarisation results, which is explained in experiments and results Section 6.4.

6.4 Experiments and results

In this Section we explain the different setups for experiments and summary evaluation on two datasets –news domain dataset from DUC corpus and Medical domain dataset.

6.4.1 Setup

We have considered two datasets of different genre for our experiments. First is DUC2002 dataset. It consists of total 565 documents divided into 59 topics, out of which 537 are unique

documents and 28 documents are repeated under different topics. These documents are news articles about current affairs. The other dataset is medical domain papers. It consists of 60 papers extracted from various medical domain journals. This corpus has been used in abstractive summarisation[14].

As described earlier, in order to analyse the effect of various steps in summary generation from O-O semantic graph we have experimented by including frequent verb-objects as object in the O-O semantic graph (*FrequentVerbObject_Included*) and by excluding them (*FrequentVerbObject_excluded*). Then we have varied inclusion of operations into summary as described in Section 6.3.2 by giving equal importance to operations and paths(*(operation as path*)) and by considering operations less informative than path(*operation After path*). We have varied edit distance threshold from 1 to 5 to remove redundant sentences from the summary, which is explained in detail in Section 6.3.3. Summary evaluation is performed by comparing the summaries generated from our different setups with the human written summaries provided by DUC committee. We have used the standard evaluation toolkit to compare the ROUGE n-gram matching results. Rouge setting used is stemmed words and no stop words included. Table 6.1 presents the results for herein stated setups.

6.4.2 ROUGE score analysis on DUC dataset

In Table 6.1 we can see that the scores Rouge-1, Rouge-2 and Rouge-L for the experimental setup *Operations ranked As paths*, where operations are ranked similar to paths, are constantly better than the experimental setup *Operations ranked After paths* where operations are ranked lower than paths.

Table 6.1: Rouge scores on DUC2002 data(stemmed words and no stopwords included).

Ranking	O-O Semantic graph	Redundancy distance	Rouge-1	Rouge-2	Rouge-L
Operations ranked As paths	*FrequentVerbObject_Excluded*	0	0.37174	0.11059	0.34939
-	-	1	0.37396	0.11192	0.35201
-	-	2	0.37718	0.11292	0.35491
-	-	3	0.37735	0.1131	0.35475
-	-	4	0.37383	**0.11355**	0.34973
-	-	5	0.35884	0.10832	0.33458
Operations ranked As paths	*FrequentVerbObject_Included*	0	0.37227	0.11122	0.3505
-	-	1	0.3746	0.11213	0.3528
-	-	2	**0.37764**	0.11301	**0.35563**
-	-	3	0.37754	0.11342	0.35493
-	-	4	0.37426	0.11349	0.35032
-	-	5	0.35917	0.10885	0.33512
Operations ranked After paths	*FrequentVerbObject_Excluded*	0	0.36712	0.10635	0.34459
-	-	1	0.36856	0.10717	0.34514
-	-	2	0.37049	0.1081	0.347
-	-	3	0.37303	0.10896	0.34917
-	-	4	0.37224	0.10997	0.34684
Operations ranked After paths	*FrequentVerbObject_Included*	0	0.3668	0.10598	0.34443
-	-	1	0.36829	0.10701	0.34472
-	-	2	0.37023	0.10822	0.34687
-	-	3	0.37298	0.10887	0.34909
-	-	4	0.37256	0.11004	0.34709

Operations generally involve one entity and a verb, whereas a smallest path is made of two entities connected by a relation. The score indicates that a summary should give equal

importance to interactive sentences between entities and to the descriptive sentences, which does not involve more than one entity.

The difference of results in including frequent verb-object(nouns) as objects in O-O semantic graph can be observed from Table 6.1. We see that in the experimental setup *Operations ranked After paths* along with *FrequentVerbObject_Included*, it does not bring any improvement. But for the best performing setup *Operations ranked As paths*, it brings improvement to include frequent verb-objects in the O-O semantic graph. So far the best results on DUC data has been achieved by *Operations ranked As paths* along with *FrequentVerbObject_Included* and by keeping redundancy distance 2.

6.4.3 Impact of adding adjectives/adverbs in the summary

In object-oriented semantic graph, adjectives become property of object and in cases where the noun does not become an object but become argument of operation the adjective is added along with the noun in the operation itself. Summaries generated from O-O semantic graph already contain around half of the adjectives which are part of operations and properties of relations. As earlier explained in Section 6.2.3, the development of ranking methods for properties of objects is left as future work and we do not include properties of objects in summary. It leaves out those adjectives to be added in summary which may be connected to objects and becomes their property.

To analyse the effect of including adjectives in summary we first manually inspected a few summary sentences. We observed that in some cases adjectives are required to unambiguously present the intended meaning of sentence but in other cases excluding adjectives from sentence doesn't affect its meaning. For example in the generated summary sentence *Warning posted for Haiti, Cuba and Cayman Islands.* if we add the identified

properties from adjectives (*small , british*) to object (Cayman Islands), the sentence becomes *Warning posted for Haiti, Cuba and small british Cayman Islands.* In the above discussed sentence, the entity *Cayman Islands* is clearly recognisable even without any adjective and adding the adjectives about demographic and geographic properties does not make any difference to the intended meaning. On the contrary the quantifying adjectives are essential to present the unambiguous meaning of sentence. For example consider the sentence *Jamaicans stayed home to prepare for storm*. If we add the adjective *few* to the given sentence it will change the meaning of sentence as shown below.

Few Jamaicans stayed home to prepare for storm.

From looking at the summaries produced by human, we have seen that some authors leave out the adjectives from the well identified named entities. Most kept adjectives are about quantities (most/few), parts (eastern, western), intensity (heavy, strong) or cardinality (i.e. one, two). These adjective helps in clearly separating that particular instance of information from other similar concepts.

In our approach of summary generation from O-O semantic graph we are already keeping numerical information in the summary taken from cardinal dependency relation. Our current approach contains around 50% of the adjectives. To understand the effect of adjectives/adverbs clearly we decided to do three different summarisation experiments: i.*without any adjective/adverb*, ii.*with adjective only* and iii.*with both adjective and adverbs*. The results are shown in Table 6.2.

Table 6.2: Impact on Rouge scores after adding adjective/adverbs in summary.

	ROUGE-1	ROUGE-2	ROUGE-L
Without Adjective/adverb	0.36726	0.10344	0.34639
Current approach(~50% adjectives)	0.37764	0.11342	0.35493
With all adjectives possible to add	**0.37891**	**0.11669**	**0.35486**
with adjectives and adverbs possible to add	0.37435	0.11443	0.35042

We can see that adding all possible adjectives performs best, although the difference from our earlier approach, where only part of the adjectives were included, is not much significant. Comparing the results *without adjectives/adverb* (0.36726) to the results *with adjective* (0.37891) we observe 1% improvement in the latter approach. It indicates that that a summary with adjective words included is more informative and more similar to human written summary than the summary without any adjective words.

We have also added adverbs to the summary with all adjectives, but the scores were reduced. Adverbs describe about quality of a verb or adjective. It also describes the temporal information about the verb and this has been already added to our current approach by checking time modifier dependency relations. So the best approach for summarisation can be including all adjectives along with temporal modifier adverbs.

ROUGE evaluations generally considers summaries to be of 100 words length and in the DUC summarisation corpus the human written summaries are of 100 words length for comparison with the system generated summaries. In our abstractive summarisation approach, when the O-O semantic graph has less relations and operations the system generated summary length couldn't reach 100 words. To make a fair comparison between system generated and human written summaries for ROUGE evaluation we added the

properties of objects to fulfil the words limit wherever summaries were shorter than 100 words. The scores after this modification in system generated summaries are shown in Table 6.3.

Table 6.3: Rouge score on DUC2002 corpus for maximum100 word summary

	ROUGE-1	ROUGE-2	ROUGE-L
	0.38142	0.11748	0.35631

6.4.4 Comparison of ROUGE score with other summarisers

We here describe the comparison of our best results with the results from other available summarisers. We have taken *dense semantic graph based summariser* as our baseline, because O-O semantic graph is an extension of dense semantic graph and we want to compare the performance improvement made by it. We have also compared with the summaries generated by openly available summariser- *open text summariser (OTS*). We have cited the results available in journal articles of the best participants in DUC-2002 task and LSA based summarisers [137] to compare with our results. We have chosen their results as they have quoted their results on the same setting (no stop word, all words converted to stem words and 100 words summary). Since in our summarisation we were already losing many prepositional connectors (considered as stopwords), we decided to compare summaries on the setting without stopwords for a fair comparison of generated summaries. Results are shown in Table 6.4.

A short description of summarisers participating in DUC2002 is given here. Summarisation system 28 [97] uses a HMM classifier based extraction approach trained on a feature set made of position of sentence, query term identification and previous sentence selection for

summary. This was the best performing system in DUC-2002 summarisation task. System 21 [219] was an approach based on Weighted Probability Distribution Voting which was trained on a feature sets based on frequency, distance between occurrences of tokens, positions of sentences and it was used to recognise writing style of authors and to extract sentences that author may consider important. System 31 [220] was unsupervised extraction method based on topic selection by scoring segments and lexical chains. In summarisation approach of System 29 [221] layered topic segmentation is used for abstract generation, topic segments are selected for inclusion as summary depending on the summary length. All preposition words gets discarded. System 27 [98] was an extractive approach based on SVM classifier. Feature set used is position of sentences, length of sentence, frequency based weights, title words similarity score. System 15 [222] uses decision tree classifier to decide thematic segmentation of document based on number of noun phrases and other document level features. After thematic segmentation it uses key phrase matching to extract important sentences from each segment to generate the summary. System 23 [223] uses BM25 and SVM classifier and features-sentence position, count of sentences, indicators as first or last sentence, lexical links and bonds between sentences to generate the summary. System 16 [224] also popularly known as MEAD summariser follows a centroid based approach to cluster sentences around centroids and then select sentences which are closer to the centre of cluster. It also takes into account the sentence position and cosine similarity to remove redundancy. In System 18 [225] a Bayesian classifier is trained on features word count, enclosing XML element tag, position of enclosing paragraph in document , sentence position in the paragraph, frequency count of words and similarity to headline words. Gate's Annie anaphora resolution system is used to resolve dangling references in the generated summaries. System 25 [226] uses a four step approach to first map the document to a canonical database to explore the main entities and their relation in document. A collection

classifier decide from the linguistic feature the type of document from four option ex. Related to a person or an event and then generated headlines around the type recognised for it. System 17 [227] generates indicative-informative summaries based on linguistic features. System 30 [228] uses noisy channel model to generate headline style summaries. Source model used for this is bigram probabilities of headline words used in the stories taken from well know news archives.

Dense semantic graph [229], our baseline summariser, is described in detail in Chapter 4. It's a semantic graph based extractive approach.

Latent Semantic Analysis based summarisers, GLLSA, LeSA, LeSA+AR [137], have been popular semantic representation based summarisers. LSA technique identifies the hidden topics in the document by calculating the left singular eigen vectors of the term-sentence frequency matrix of the document. It also decides the number of important topics to be covered in summary by reducing the dimension of topics by applying Singular Vector Decomposition. GLLSA is purely based on LSA to extract the summary sentences after applying SVD. LeLSA is improvement over GLLSA method by incorporating length of sentences as additional feature. LeLSA+AR improve the LeLSA method by incorporating anaphora resolution to produce more coherent summaries.

OTS is an open source text summariser. It is found to be best performing among all openly available text summarisers on the web. It is frequency based and incorporates a lexicon to derive synonymy relations. It also takes into account the usage of clue words to decide the importance of sentence to perform extractive summarisation.

We can see in Table 6.4 that the abstractive approach based on O-O semantic graph has achieved better Rouge-1 and Rouge-L scores than the extractive approach of dense semantic graph based summarisation. It signifies the improvements made in summary generation after

interpreting and representing information in terms of composite units that are *objects* rather than the atomic units *triples* which were the basic building blocks of dense semantic graph. In addition, it shows the credibility of the algorithm proposed to generate summary from O-O semantic graph representation of text document.

Table 6.4: Comparison of different summarisers on DUC2002 corpus.

System	ROUGE-1	ROUGE-2	ROUGE-L
28	0.42776	0.21769	0.38645
LeLSA+AR[1]	0.42280	0.20741	0.39276
21	0.41488	0.21038	0.37543
DUC baseline	0.41132	0.21075	0.37535
19	0.40823	0.20878	0.37351
LeLSA[1]	0.40805	0.19722	0.37878
27	0.40522	0.2022	0.36913
29	0.39925	0.20057	0.36165
31	0.39457	0.19049	0.35935
15	0.38884	0.18578	0.35366
Open Text Summariser (OTS)	0.38864	0.18966	0.34388
O-O semantic graph	0.38142	0.11748	0.35631
23	0.38079	0.19587	0.34427
GLLSA[1]	0.38068	0.1744	0.35118
16	0.37147	0.17237	0.33224
Dense Semantic Graph	0.37919	0.168	0.35206
18	0.36816	0.17872	0.331
25	0.34297	0.15256	0.31056
Random	0.29963	0.11095	0.27951
17	0.13528	0.0569	0.12193
30	0.07452	0.03745	0.06985

Rouge-L score which is count of the common longest sequences between two summaries shows the sentence level similarity between them. Rouge-L scores of different summarisers listed in Table 6.4 shows that our abstractive approach based on O-O semantic graph performs better than extractive summarisation by latent semantic analysis approach- *GLLSA*, dense semantic graph and the benchmark OTS summariser in sentence level similarity comparison to human written summaries. In Rouge-1 score our approach performed better than *GLLSA* and many other DUC participants (System Id 23, 16, 18, 25, 17 and 30) and is close to performances of benchmark system OTS. Although our approach couldn't reach the performances of top ranking systems in DUC (System 28, 21, 19, 27, 29, 31) and the LeSA + AR approach.

One reason of this average performance can be that our summaries are directly generated from the O-O semantic graph representation of text documents and in the n-gram match score of Rouge evaluation it is difficult to surpass the summarisers which are extracting original well-formed sentences from the documents. Also at 1-gram matching level our scores are quite low compared to top performing systems, one reason being that in our case all pronouns gets replaced with proper references due to our pre-processing step of resolving pronominal references and thus we lose the score for matching pronominal words. But still its comparative performance with the good performing summarisers suggests that abstractive summarisation from representation of text document is an achievable goal and O-O semantic graph representation can form the basis for it.

6.4.5 LDA similarity based evaluation of topic coverage

ROUGE metric suffers from the disagreement in reference summaries written by different human authors when only few reference summaries are available. To overcome the

evaluation issues due to scarcity of gold standard summaries new measures have been proposed to compare summaries with the original document. There are recent works on using topic modelling techniques[161] to measure topic similarity between original document and the system generated summary as an evaluation metric for text summarisation. It gives scope of extending the summarisation corpus to the domains where we don't have human written summaries. We have used one of topic modelling techniques *LDA* to compare the probability distribution of system generated summaries with the original documents. LDA considers every document as mixture of various topics[57]. It models the documents as a set of topics with probability distribution $P(t_j|D_A)$ and models topics as set of words with probability distribution $P(w_i|t_j)$. The probability distribution of topic t_i in document D_A represents the coverage of t_i in D_A. The word distribution w_i in topic t_i indicates the importance of word w_i to topic t_i. We have estimated the LDA model from original 533 documents of DUC-2002 summarisation corpus after removing stop words and converting all words to base form using Stanford's stemmer.

Using this model the probability distributions for original document and for the summaries generated by systems are inferenced. We have utilised SEMILAR semantic library [230] for probability estimation and inference. After inference we compared the document distributions over topics in system summaries with the original documents by Information Radius similarity measure (IR) also known as Jensen-Shannon divergence between two probability distributions[231]. We perform this comparison for each unique document in DUC-2002 corpus. For each document we have 3 different system generated summaries generated from O-O semantic graph summariser, dense semantic graph summariser and open text summariser to compare with the original document. After calculating IR value for each pair of summary/document we sum up all the IR values for each summariser and then average it by

dividing the count of documents. Table 6.5 shows the averages of IR similarity for the summaries generated from each summariser.

Table 6.5 : Average LDA based similarity of summaries with original documents.

	Dense semantic graph	O-O Semantic Graph	Open Text Summariser
Average IR	0.949712789	0.956508037	0.946131807

In Table 6.5 we can see that although difference is not very large among the average IR values of all summarisers, yet O-O semantic graph based summariser has the better average IR value among all. It signifies that the topic distribution in summaries by O-O semantic graph summariser is the closest to the topic distribution in original documents among all other summaries. It shows the usefulness of abstractive summarisation from representation over extractive summarisation.

6.4.6 Rouge score analysis on medical dataset

We have also experimented on another data set from a very different genre: Medical domain. We wanted to compare against some abstractive summariser and this corpus was used in compendium abstractive summarisation [14]. We have used the exact Rouge setting provided in the paper, which has word length 162 words, stemmed and no stop word. No changes were made to our abstractive summariser to incorporate any domain specific knowledge.

Table 6.6 shows the results of text summarisation on medical domain dataset from our approach on summary generation from O-O semantic graph. The results shown here are best results obtained by varying setting of graph as described from previous corpus on DUC2002. Results are shown by varying redundancy removal level and effect of inclusion or exclusion of frequent verb objects in the semantic graph. Edit distance for redundancy removal is varied

from 0-7. Improvements in the results were seen from edit distance 3 to edit distance 7. Compared to previous corpus where optimum results were found at less redundancy level, we looked through the generated summaries to understand the reason. It was found that medical domain corpus has long names for medicines or different components involved and repeating named of these entities causes much unwanted redundant information to float through summaries. A reference taken from this corpus is "*Bio Rad Laboratories bioplex"* which is present in two sentences a. *Bio Rad Laboratories bioplex uses Vasculitis kit.* b. *Bio Rad Laboratories bioplex differs from other method.*

By increasing the edit distance these repetitive sentences are getting eliminated by giving way to more diverse information. However fusion of sentences or simply generating appropriate small pronoun references can help preventing the redundancy, although it will require determination of exact reference gender to devise suitable pronoun and can be a further research area. In Table 6.6 we can also see that inclusion of frequent verb-objects as object in the semantic graph deteriorate the performance a bit for same redundancy levels.

Table 6.6: Rouge scores on Medical data (word length 162 words, stemmed and no stop word included)

O-O Semantic graph	Redundancy distance	Rouge-1	Rouge-2	Rouge-L
FrequentVerbObject_Excluded	3	0.29202	0.05249	0.27427
-	4	0.29394	0.05407	0.27431
-	5	0.30009	0.05407	0.27921
-	6	0.30277	0.05576	0.2812
-	7	0.31036	0.05818	0.28489
FrequentVerbObject_Included	3	0.28863	0.05098	0.26850
-	4	0.29343	0.05145	0.27397
-	5	0.29778	0.05239	0.27798
-	6	0.29907	0.05273	0.27823
-	7	0.30409	0.05326	0.28056

We have also compared our system performance with compendium abstractive summariser on this dataset. Results for compendium summariser for same rouge setting are quoted from the published paper [14]. Rouge-1 and Rouge-2 scores for O-O semantic graph based summariser are lesser than compendium summariser as shown in Table 6.7. The Rouge-L score of the O-O semantic graph based summariser has outperformed compendium summariser Rouge-L score. Rouge-L score is the longest sequence match between the sentences of the system summaries and human written summaries. Better Rouge-L score indicates better sentence level similarity. It indicates good performance of our summariser in other domains than only news domain, for which most of the extractive summarisers have been designed. It can be further improved for medical domain by utilizing medical domain ontology (UMLS), instead of general WordNet ontology that has been currently used in our summariser.

Table 6.7: Rouge scores comparison on medical domain corpus.

Systems	Rouge-1	Rouge-2	Rouge-L
compendium*E–A*[14]	38.66	11.49	25.95
Object-oriented Semantic Graph summariser	31.036	5.818	28.489

6.5 Conclusion

In this Chapter we described our abstractive single document summarisation approach which generates summary from representation of text directly. Text representation scheme used was object-oriented semantic graph. It is a semantic graph representation to view text document as an interaction of objects. Implementation details of the construction of object-oriented semantic graph are provided in Chapter 5. In current Chapter we have described the

methodologies to generate summary from the semantic graph. The summarisation methods were evaluated on different datasets by two evaluation measures (Rouge toolkit and LDA based similarity measure). Overall results indicate that abstractive summarization presented in this paper is comparable with the extractive summarisation based on other semantic graphs. This approach is among the few novel researches done for summary generation from the representation of text instead of original text. So the comparable results are encouraging to continue more research in this direction.

Chapter 7

Conclusion

7.1 Thesis contribution

The aim of this research was to analyse semantic properties of text documents and contribute in design and development of an efficient semantic representation and abstractive summarisation system for text documents. An efficient semantic representation is useful in articulating the meaning of text and the context into a structured form. It enables application of machine learning algorithms (i.e. clustering) and graph ranking algorithms on textual data corpuses for various NLP tasks i.e. text summarisation. In this research we have proposed a novel semantic representation-*Object oriented semantic graph* which models the text document into a connected graph of objects, properties and relations. We have designed rule based system to construct the *object-oriented semantic graph* from a text document written in natural language. Further research on the application of *O-O semantic graph* resulted in a new abstractive summarisation system. Through experiments on standard summarisation corpuses we have compared the performance of our abstractive summarisation system with other semantic graph based summarisation systems and found the performance comparable. We list the contributions of this thesis here.

1. In Chapter 3 we have proposed better ways to align sentences. We have also tested the inclusion of clauses as the alignment units and its effect on overall alignment of sentences. We have proposed a methodology to reduce time complexity of original alignment while still preserving the intended alignment. In addition, we have shown through the experiments that

the proposed methodology is useful in reducing time cost for sentence pairs with length more than 40 words. Further extension to this work should include improved ways of clause splitting by utilizing more semantic and syntactic information. For node similarities further concept based similarity measures can be utilized to make the alignment more accurate.

2. In Chapter 4 we have researched two ways to construct semantic graphs of textual information from dependency relations and their application in extractive text summarisation. We contributed to knowledge by developing two approaches: the *triple based semantic graph* and the *dense semantic graph* using the open source JUNG library and CoreNLP toolkit. Summarisers were developed based on these semantic graphs and their *PageRank* analysis. Our experiments on these summarisers for standard DUC corpora has contributed the knowledge that that the more the dependency relations are included in the semantic graph generation the more accurate are the *PageRank* scores of semantic nodes and thus the more accurate are the rankings of the sentences for text summarisation. Also by including sentence level features, such as sentence position, the summarisation scores improved and thus the implemented summarisers are open for further improvements by including more semantic and syntactic features. These summarisers can be further investigated by using the contributed implementations. In future work on this topic, semantic similarity measure and word sense disambiguation can be applied to improve the connectivity in the *dense semantic graph* by to identifying more relations between nodes. Dense semantic graphs can be improved to be more visually understandable and more efficient for direct abstractive summary generation instead of extracting from the original document.

3. In Chapter 5 we have looked into the different ways of modelling natural language text and the various semantic graphs used in natural language processing. We have analysed the complexities in sentence formation by different writers that make modelling textual content a difficult task. Sentences can vary from a simple subject and predicate to a complex sentence

of many clauses. Transforming textual content to a graphical representation which preserves the semantics and modularises information into clear non-ambiguous subparts is an ambitious goal. We have proposed our solution in the form of an object-oriented semantic graph and have provided detailed rules and their implementations to generate the graph. This graph can be utilised for various NLP task. In the following Chapter we contributed the use of this proposed graph in the text summarisation task.

4. In our last study we have worked on abstractive summarisation. Our approach helps in bringing the text to text generation methods such as sentence compression, fusion and enhancement to single document summarisation where these approaches are already successfully used in multi-document summarisation. Usage of sentence fusion and enhancement is popular in multi-document summarisation because multiple documents about the same topic provide as input various similar sentences and redundant information to be fused together. Our work was focused on single document summarisation. Our approach of summarisation was based on a semantic representation of information that is an *object-oriented semantic graph*. In this approach sentences are divided into smaller semantic units and this division can be thought of as similar to sentence simplification. This gave a better way to extract information without any information loss or ambiguity of information. Also all similar events (predicates) were identified by a unique sentence number and word number combination to avoid merging wrong information in the summary. Information gets fused as adjectives/appositives from different sentences about the same entity and merged into one sentence, even though the source sentences are no longer in the summary. Here we differ from other sentence fusion techniques in that we perform deeper semantic analysis of whether the information can be considered as an independent property of the entity and only then we merge it. We merge or fuse sentences at the noun level, where information about the

entity from various sentences is combined under one Object, in the form of properties or operations.

Comparison between extractive summarisation based on *dense semantic graph* representation and abstractive summarisation from the *object-oriented semantic graph* shows that abstraction performs as well as extraction using ROUGE measures. It has better sentence level similarity score ROUGE-L(longest common sequence match) than the extractive summarisation. Generally extractive summaries lack a connected story as they may have missing references and may not contain diversity as bigger extracted sentences may leave less space for more diversified information to be added. Our abstractive summarizer is a step towards reducing the limitation imposed by extractive summarisation. We resolve co-references at the beginning when constructing the graph representation and information is divided into smaller semantic units to give way for adding more diversified information in the summary.

Comparison with state of the art methods of summarisation, which are again extractive in nature, shows that our abstractive approach performs lower than the best summarisers do. This can be due to the fact that sentences taken from original document have more chance of scoring better in n-gram matching evaluation of ROUGE evaluation toolkit. As in our summary all the pronoun information gets converted to entity names, we have tried to compare summaries at ROUGE setting which removes *stopwords* that includes pronouns. Due to replacement of entity name for pronoun at many places our summary has longer entity names repeated in the same sentence, which can be resolved to smaller pronouns if natural language generation strategies are improved. It will make space to include more summary words in the limited length summaries.

This can also lead to a good hybrid approach of combining extraction with abstraction. So the summaries from extractive summarisers can be given as input to abstractive summarisers and

then we can get a better summary which is more diverse and coherent. Given this analysis, we have shown the value of using object-oriented semantic graphs for summarisation and its future prospects in our last research study. Also summary generation from an efficient representation gives control over level of summary generation. Depending on the summary length, information to be added can be varied from only object and relations to objects, properties, operations and relations. More work can be done on raking of properties to decide their inclusion in the summary. We do not merge sentences based on predicate, which could be done by including some good performing event co-reference resolver.

Overall our research has contributed in terms of analysing different semantic representations of text documents for automated text summarisation. The major contribution is development of graph generator for object-oriented semantic graph construction from any text document and summariser for abstractive summary generation.

7.2 Future work

We aim to continue the research in this direction to further improve the graph generator to handle complexities of natural language. In following ways this work can be research further.

7.2.1 Enhancing the graph generator for complex sentences

Presently the graph generator can handle object identification from different subordinate clauses in declarative sentences but its generalised version should be able to work with imperative, interrogative and exclamatory type of sentences. New rules should be formed by analysing different type of sentences and their difference from declarative type of sentences. Graph generator should also have better sophisticated strategies to decide whether mass nouns or count nouns should be made objects depending on their importance in the text.

Presently the synonymy check is limited to noun level which can be enhanced to verb level and based on this relations can be merged. This will avoid repetitive relations in the graph and will make ranking more efficient. If improvement can be made in anaphora resolution system then it will avoid misrepresentation of information when wrong entities are replaced for the references.

7.2.2 Enhancing the graph generator for other languages

We should port the graph generator to the available *universal dependency relation set*[177]. Universal dependencies are a set of universal grammatical relation which can be used to capture relations between words in any language. Presently all rules are designed to follow the Stanford's dependency convention for English language. Rules should be modified to incorporate the universal dependency relations so it can be available to be utilised in other languages as well.

7.2.3 Improving summariser

Immediate scope of improvement in abstractive summariser is designing of methodology to rank properties of objects and their inclusion in text summaries. This can be achieved by designing methodology which utilises conceptual similarity of property to the object. Also better natural language generation capabilities should be included to generate correct grammatical summary.

References

[1] J. M. Perea-Ortega, E. Lloret, L. Alfonso Ureña-López, and M. Palomar, "Application of Text Summarization techniques to the Geographical Information Retrieval task," *Expert Syst. Appl.*, vol. 40, no. 8, pp. 2966–2974, Jun. 2013.

[2] J. C. Feblowitz, A. Wright, H. Singh, L. Samal, and D. F. Sittig, "Summarization of clinical information: a conceptual model.," *J. Biomed. Inform.*, vol. 44, no. 4, pp. 688–99, Aug. 2011.

[3] N. Elhadad and K. R. McKeown, "Towards Generating Patient Specific Summaries of Medical Articles," in *Proceedings of the NAACL'2001 Workshop on Automatic Summarization*, 2001.

[4] N. Elhadad, "User-sensitive text summarization thesis summary," in *Proceedings of the 19th national conference on Artifical intelligence*, 2004, pp. 987–988.

[5] E. Lloret, H. Llorens, P. Moreda, E. Saquete, and M. Palomar, "Text Summarization Contribution to Semantic Question Answering: New Approaches for Finding Answers on the Web," *Int. J. Intell. Syst.*, vol. 26, no. 12, pp. 1125–1152, 2014.

[6] S. Chakraborty and L. Subramanian, "Location specific summarization of climatic and agricultural trends," in *Proceedings of the 20th international conference companion on World wide web - WWW '11*, 2011, p. 463.

[7] S. Corston-oliver, "Text compaction for display on very small screens," in *Proceedings of the NAACL-2001*, 2001, pp. 89–98.

[8] E. Lloret and M. Palomar, "Towards automatic tweet generation: A comparative study from the text summarization perspective in the journalism genre," *Expert Syst. Appl.*, vol. 40, no. 16, pp. 6624–6630, Nov. 2013.

[9] K. S. Jones, *Automatic summarising : factors and directions, Advances in automatic text summarization*. MIT Press, 1998.

[10] T. L. Griffiths, M. Steyvers, and J. B. Tenenbaum, "Topics in semantic representation.," *Psychol. Rev.*, vol. 114, no. 2, pp. 211–44, Apr. 2007.

[11] P. E. Genest, G. Lapalme, and M. Yousfi-Monod, "HexTac: the creation of a manual extractive run," in *Proceedings of the Second Text Analysis Conference*, 2010, p. 6.

[12] E. Lloret and M. Palomar, "Quantifying the Limits and Success of Extractive Summarization Systems Across Domains," in *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, 2010, no. June, pp. 903–911.

[13] F. Liu and Y. Liu, "From extractive to abstractive meeting summaries: can it be done by sentence compression?," in *Proceedings of the ACL-IJCNLP 2009 Conference Short Papers*, 2009, pp. 261–264.

[14] E. Lloret, M. T. Romá-Ferri, and M. Palomar, "COMPENDIUM: A text summarization system for generating abstracts of research papers," *Data Knowl. Eng.*, vol. 88, pp. 164–175, Nov. 2013.

[15] G. Carenini, J. C. K. Cheung, and A. Pauls, "Multi-document summarization of evaluative text," *Comput. Intell.*, vol. 29, no. 4, pp. 545–576, 2013.

[16] J. C. K. Cheung, "Comparing Abstractive and Extractive Summarization of Evaluative Text : Controversiality and Content Selection," 2008.

[17] E. Lloret and M. Palomar, "Text summarisation in progress: a literature review," *Artif. Intell. Rev.*, vol. 37, no. 1, pp. 1–41, Apr. 2012.

[18] P. M. Nadkarni, L. Ohno-Machado, and W. W. Chapman, “Natural language processing: an introduction,” *J. Am. Med. Informatics Assoc.*, vol. 18, no. 5, pp. 544–551, 2011.

[19] D. Klein and C. D. Manning, “Accurate Unlexicalized Parsing,” in *Proceedings of the 41st Meeting of the Association for Computational Linguistics*, 2003, pp. 423–430.

[20] Z. Zhong and H. T. Ng, “It Makes Sense: A wide-coverage word sense disambiguation system for free text,” *Proc. ACL 2010 (System Demonstr.*, pp. 78–83, 2010.

[21] E. Agirre, O. L. de Lacalle, and A. Soroa, “Random Walks for Knowledge-Based Word Sense Disambiguation,” *Comput. Linguist.*, vol. 40, no. 1, pp. 57–84, Mar. 2014.

[22] L. Plaza and A. Díaz, “Using Semantic Graphs and Word Sense Disambiguation Techniques to Improve Text Summarization,” *Proces. del Leng. Nat. Rev.*, vol. 47, pp. 97–105, 2011.

[23] D. Rusu, B. Fortuna, and D. Mladenić, “Improved Semantic Graphs with Word Sense Disambiguation,” in *The Fifth International Conference on Knowledge Capture*, 2009.

[24] M. De Marneffe, B. Maccartney, and C. D. Manning, “Generating Typed Dependency Parses from Phrase Structure Parses,” in *LREC 2006*, 2006.

[25] J. Nivre, J. Hall, and J. Nilsson, “MaltParser: A data-driven parser-generator for dependency parsing,” in *Proceedings of LREC*, 2006, vol. 6, pp. 2216–2219.

[26] D. Nadeau and S. Sekine, “A survey of named entity recognition and classification,” *Linguist. Investig.*, no. 30, pp. 3–26., 2007.

[27] L. Ratinov and D. Roth, “Design challenges and misconceptions in named entity recognition,” *Proc. Thirteen. Conf. Comput. Nat. Lang. Learn. CoNLL 09*, no. June, p. 147, 2009.

[28] L. Ratinov, D. Roth, D. Downey, and M. Anderson, “Local and global algorithms for disambiguation to Wikipedia,” in *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics*, 2011.

[29] J. R. Finkel and C. D. Manning, “Joint parsing and named entity recognition,” *Proc. Hum. Lang. Technol. 2009 Annu. Conf. North Am. Chapter Assoc. Comput. Linguist. - NAACL ’09*, pp. 326–334, 2009.

[30] H. Lee, Y. Peirsman, A. Chang, N. Chambers, M. Surdeanu, and D. Jurafsky, “Stanford’s multi-pass sieve coreference resolution system at the CoNLL-2011 shared task,” in *Proceedings of the CoNLL-2011 Shared Task*, 2011, pp. 28–34.

[31] E. R. Fernandes, C. N. dos Santos, and R. L. Milidiú, “Latent structure perceptron with feature induction for unrestricted coreference resolution,” in *Joint Conference on EMNLP and CoNLL-Shared task*, 2012, pp. 41–48.

[32] K.-W. Chang, R. Samdani, and D. Roth, “A Constrained Latent Variable Model for Coreference Resolution,” in *Emnlp*, 2013, vol. 2011, no. 2010.

[33] H. Hajishirzi, L. Zilles, and D. S. Weld, “Joint Coreference Resolution and Named-Entity Linking with Multi-pass Sieves,” in *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, 2012, no. July, pp. 489–500.

[34] X. Yao, B. Van Durme, C. Callison-Burch, and P. Clark, “A Lightweight and High Performance Monolingual Word Aligner,” *Proc. 51st Annu. Meet. Assoc. Comput. Linguist.*, pp. 702–707, 2013.

[35] B. MacCartney, M. Galley, and C. D. Manning, “A phrase-based alignment model for natural language inference,” *Proc. EMNLP*, p. 802, 2008.

[36] K. Thadani and K. Mckeown, “Optimal and Syntactically-Informed Decoding for Monolingual Phrase-Based Alignment,” *Comput. Linguist.*, pp. 254–259, 2011.

[37] I. Androutsopoulos and P. Malakasiotis, “A Survey of Paraphrasing and Textual Entailment Methods,” no. 5, pp. 1–53, 2010.

[38] E. Lloret, Ó. Ferrández, R. Muñoz, and M. Palomar, “A Text Summarization Approach under the Influence of Textual Entailment,” in *Proceedings of the 5th International Workshop on Natural Language Processing and Cognitive Science (NLPCS 2008)*, 2008, pp. 22–31.

[39] M. Rios and A. Gelbukh, “Recognizing textual entailment with a semantic edit distance metric,” in *Proceedings of Special Session - Revised Papers, 11th Mexican International Conference on Artificial Intelligence 2012: Advances in Artificial Intelligence and Applications, MICAI 2012*, 2012, pp. 15–20.

[40] N. Sharma, R. Sharma, and K. K. Biswas, *Recognizing Textual Entailment using Dependency Analysis and Machine Learning*. 2015.

[41] H. Daumé III and D. Marcu, “A Tree-Position Kernel for Document Compression,” in *Proceedings of DUC2004*, 2004.

[42] R. Barzilay and M. Elhadad, “Using Lexical Chains for Text Summarization,” in *Proceedings of ACL Workshop on Intelligent Scalable Text Summarization*, 1997.

[43] K. R. McKeown, J. L. Klavans, V. Hatzivassiloglou, R. Barzilay, and E. Eskin, “Towards multidocument summarization by reformulation: progress and prospects,” in *Proceeding of the 16th national conference of the American association for artificial intelligence*, 1999.

[44] D. R. Radev, H. Jing, M. Styś, and D. Tam, “Centroid-based summarization of multiple documents,” *Inf. Process. Manag.*, vol. 40, no. 6, pp. 919–938, Nov. 2004.

[45] R. Barzilay and K. R. Mckeown, “Sentence Fusion for Multidocument News Summarization,” *Comput. Linguist.*, vol. 31, no. 3, pp. 297–328, 2005.

[46] C. D. Paice, “Constructing literature abstracts by computer: Techniques and prospects,” *Inf. Process. Manag.*, vol. 26, no. 1, pp. 171–186, Jan. 1990.

[47] H. Daumé III and D. Marcu, “Bayesian query-focused summarization,” in *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the ACL - ACL ’06*, 2006.

[48] I. Mani, “Recent developments in text summarization,” in *Proceedings of the tenth international conference on Information and knowledge management - CIKM’01*, 2001.

[49] M.-Y. Kan, K. R. McKeown, and J. L. Klavans, “Applying Natural Language Generation to Indicative Summarization,” in *Proceedings of the 8th European workshop on Natural Language Generation*, 2001.

[50] M. Hu and B. Liu, “Mining and summarizing customer reviews,” in *Proceedings of the 2004 ACM SIGKDD international conference on Knowledge discovery and data mining - KDD ’04*, 2004.

[51] A. Balahur, E. Lloret, Ó. Ferrández, A. Montoyo, M. Palomar, and R. Muñoz, “The DLSIUAES Team’s Participation in the TAC 2008 Tracks PART I : Opinion Summarization Pilot,” in *Proceedings of the Text Analysis Conference (TAC)*, 2008.

[52] F. Schilder, R. Kondadadi, J. L. Leidner, and J. G. Conrad, “Thomson Reuters at TAC 2008 : Aggressive Filtering with FastSum for Update and Opinion Summarization,” in *Proceedings of the Text Analysis Conference (TAC)*, 2008.

[53] C. Sauper and R. Barzilay, “Automatically generating Wikipedia articles: a structure-aware approach,” in *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, 2009.

[54] R. Yan, X. Wan, J. Otterbacher, L. Kong, X. Li, and Y. Zhang, “Evolutionary timeline

summarization," in *Proceedings of the 34th international ACM SIGIR conference on Research and development in Information - SIGIR '11*, 2011, p. 745.

[55] S. Bauer and S. Teufel, "A Methodology for Evaluating Timeline Generation Algorithms based on Deep Semantic Units," in *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, 2015, pp. 834–839.

[56] I. Matveeva, G.-A. Levow, A. Farahat, and C. Royer, *Term representation with generalized latent semantic analysis*, vol. 5. 2007.

[57] D. M. Blei, A. Y. Ng, and M. I. Jordan, "Latent Dirichlet Allocation," *J. Mach. Learn. Res.*, vol. 3, pp. 993–1022, 2003.

[58] W. C. Mann and S. A. Thompson, "Rhetoric structure theory: A theory of text organisation.Technical Report ISI/RS," California, 1987.

[59] J. Brooke, "A Syntactic and Lexical-Based Discourse Segmenter," *Ijcnlp2009*, no. August, pp. 77–80, 2009.

[60] L. Polanyi, C. Culy, M. Van Den Berg, G. L. Thione, and D. Ahn, "A rule based approach to discourse parsing," in *Proceedings of the SIGDIAL'04 Workshop*, 2004, no. Ldm, pp. 108–117.

[61] D. Marcu, *The Theory and Practice of Discourse Parsing and Summarization*. MIT Press, 2000.

[62] R. Soricut and D. Marcu, "Sentence level discourse parsing using syntactic and lexical information," in *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology - NAACL '03*, 2003, vol. 1, pp. 149–156.

[63] H. Hernault, H. Prendinger, D. a. DuVerle, and M. Ishizuka, "HILDA: A discourse parser using Support Vector Machine classification," *Dialogue & Discourse*, vol. 1, no. 3, pp. 1–33, 2010.

[64] S. Joty and R. T. Ng, "CODRA : A Novel Discriminative Framework for Rhetorical Analysis," *Comput. Linguist.*, no. January, pp. 1–50, 2015.

[65] F. Rotella, F. Leuzzi, and S. Ferilli, "Learning and exploiting concept networks with ConNeKTion," *Appl. Intell.*, vol. 42, no. 1, pp. 87–111, Jul. 2014.

[66] S. Hensman and J. Dunnion, "Automatically building conceptual graphs using VerbNet and WordNet," in *ISICT '04 Proceedings of the 2004 international symposium on Information and communication technologies*, 2004, pp. 115–120.

[67] D. Rusu, B. Fortuna, M. Grobelnik, and D. Mladenić, "Semantic Graphs Derived From Triplets With Application in Document Summarization," *Inform. J.*, 2009.

[68] D. Rusu, L. Dali, B. Fortuna, M. Grobelnik, and D. Mladenić, "Triplet extraction from sentences."

[69] L. Banarescu, C. Bonial, S. Cai, M. Georgescu, K. Griffitt, U. Hermjakob, K. Knight, P. Koehn, M. Palmer, and N. Schneider, "Abstract meaning representation for sembanking," in *Proceedings of the 7th Linguistic Annotation Workshop and Interoperability with Discourse*, 2013, pp. 178–186.

[70] L. Vanderwende, A. Menezes, and C. Quirk, "An AMR parser for English , French , German , Spanish and Japanese and a new AMR-annotated corpus," in *Naacl2015*, 2015, pp. 26–30.

[71] J. Flanigan, S. Thomson, J. Carbonell, C. Dyer, and N. a Smith, "A Discriminative Graph-Based Parser for the Abstract Meaning Representation," *Acl*, pp. 1426–1436, 2014.

[72] M. Pust, U. Hermjakob, K. Knight, D. Marcu, and J. May, "Using Syntax-Based Machine Translation to Parse English into Abstract Meaning Representation," *Arxiv*, no. September, pp. 1143–1154, 2015.

[73] Y. Sawai, “Semantic Structure Analysis of Noun Phrases using Abstract Meaning Representation,” in *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, 2015, pp. 851–856.

[74] F. Liu, J. Flanigan, S. Thomson, N. Sadeh, and N. A. Smith, “Toward Abstractive Summarization Using Semantic Representations,” in *NAACL HLT*, 2015.

[75] C. J. Fillmore, C. F. Baker, and R. Pickett, “Frame Semantics for Text Understanding,” in *Proceedings of WordNet and Other Lexical Resources Workshop*, 2001, pp. 59–64.

[76] A. BURCHARDT, M. PENNACCHIOTTI, S. THATER, and M. PINKAL, “Assessing the impact of frame semantics on textual entailment,” *Nat. Lang. Eng.*, vol. 15, no. Special Issue 04, pp. 527–550, 2009.

[77] D. Das, D. Chen, A. F. T. Martins, N. Schneider, and N. A. Smith, “Frame-Semantic Parsing,” *Comput. Linguist.*, vol. 40, no. 1, pp. 9–56, 2014.

[78] I. Titov and A. Klementiev, “A Bayesian approach to unsupervised semantic role induction,” in *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics EACL ’12*, 2012, pp. 12–22.

[79] A. Modi, I. Titov, and A. Klementiev, “Unsupervised induction of frame-semantic representations,” in *Proceedings of the NAACL-HLT Workshop on the Induction of Linguistic Structure WILS ’12*, 2012, pp. 1–7.

[80] G. Salton and C. Buckley, “Term-weighting approaches in automatic text retrieval,” *Inf. Process. Manag.*, vol. 24, no. 5, pp. 513–523, 1988.

[81] C. Aone, M. E. Okurowski, and J. Gorlinsky, “Trainable, scalable summarization using robust NLP and machine learning,” in *Proceedings of the 36th annual meeting on Association for Computational Linguistics*, 1998, vol. 1.

[82] J. Carbonell and J. Goldstein, “The use of MMR, diversity-based reranking for reordering documents and producing summaries,” in *Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval - SIGIR ’98*, 1998.

[83] S. Harabagiu and F. Lacatusu, “Topic Themes for Multi-Document Summarization,” in *Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval*, 2005.

[84] D. Radev, H. Jing, and M. Stys, “Centroid-based summarization of multiple documents,” *Inf. Process. &*, vol. 40, no. 6, pp. 919–938, Nov. 2004.

[85] G. A. Miller, “WordNet: A Lexical Database for English,” *Commun. ACM*, vol. 38, no. 11, pp. 39–41, 1995.

[86] K. Spärck Jones, “Automatic summarising: The state of the art,” *Inf. Process. Manag.*, vol. 43, no. 6, pp. 1449–1481, Nov. 2007.

[87] H. G. Silber and K. F. McCoy, “Efficiently Computed Lexical Chains as an Intermediate Representation for Automatic Text Summarization,” *Comput. Linguist.*, vol. 28, no. 4, pp. 487–496, Dec. 2002.

[88] T. Wei, Y. Lu, H. Chang, Q. Zhou, and X. Bao, “A semantic approach for text clustering using WordNet and lexical chains,” *Expert Syst. Appl.*, vol. 42, no. 4, pp. 2264–2275, 2015.

[89] G. Erkan and D. R. Radev, “LexRank: graph-based lexical centrality as salience in text summarization,” *J. Artif. Intell. Res.*, vol. 22, no. 1, pp. 457–479, Jul. 2004.

[90] R. Mihalcea and P. Tarau, “TextRank: Bringing Order into Texts,” in *Proceedings of Empirical Methods in Natural Language Processing*, 2004.

[91] R. Mihalcea, “Graph-based ranking algorithms for sentence extraction, applied to text summarization,” in *Proceedings of the ACL 2004 on Interactive poster and demonstration sessions* -, 2004, no. 4.

[92] J.-Y. Yeh, H.-R. Ke, and W.-P. Yang, “iSpreadRank: Ranking sentences for extraction-based summarization using feature weight propagation in the sentence similarity network,” *Expert Syst. Appl.*, vol. 35, no. 3, pp. 1451–1462, Oct. 2008.

[93] N. Kumar, K. Srinathan, and V. Varma, “A knowledge induced graph-theoretical model for extract and abstract single document summarization,” in *CICLing 2013,Lecture Notes in Computer Science*, 2013, vol. 7817, no. PART 2, pp. 408–423.

[94] M. Litvak, “Graph-Based Keyword Extraction for Single-Document Summarization,” in *MMIES ’08 Proceedings of the Workshop on Multi-source Multilingual Information Extraction and Summarization*, 2008, no. August, pp. 17–24.

[95] X. Wan, “Towards a Unified Approach to Simultaneous Single-Document and,” in *Proceeding of the 23rd international conference on computational linguistics (Coling 2010)*, 2010, no. August, pp. 1137–1145.

[96] X. Wan and J. Xiao, “Exploiting neighborhood knowledge for single document summarization and keyphrase extraction,” *ACM Trans. Inf. Syst.*, vol. 28, no. 2, pp. 1–34, May 2010.

[97] J. D. Schlesinger, M. E. Okurowski, and J. M. Conroy, “Understanding machine performance in the context of human performance for multi-document summarization,” in *DUC 2002*, 2002.

[98] T. Hirao, Y. Sasaki, H. Isozaki, and E. Maeda, “Ntt’s text summarization system for duc-2002,” in *Proceedings of the Document Understanding Conference 2002*, 2002, no. 1, pp. 104–107.

[99] J. Kupiec, J. Pedersen, and F. Chen, “A Trainable Document Summarizer,” in *Proceedings of the 18th annual international ACM-SIGIR conference on research and development in information retrieval (SIGIR 1995)*, 1995.

[100] W. T. Chuang and J. Yang, “Extracting sentence segments for text summarization,” in *Proceedings of the 23rd annual international ACM SIGIR conference on Research and development in information retrieval - SIGIR ’00*, 2000, pp. 152–159.

[101] L. Li, K. Zhou, G.-R. Xue, H. Zha, and Y. Yu, “Enhancing diversity, coverage and balance for summarization through structure learning,” in *Proceedings of the 18th international conference on World wide web - WWW ’09*, 2009, p. 71.

[102] L. Hennig, W. Umbrath, and R. Wetzker, “An Ontology-Based Approach to Text Summarization,” in *Web Intelligence and Intelligent Agent Technology, 2008. WI-IAT ’08. IEEE/WIC/ACM International Conference on*, 2008, vol. 3, pp. 291–294.

[103] M. Collins and N. Duffy, “Convolution Kernels for Natural Language,” in *Proceedings of Neural Information Processing Systems*, 2001.

[104] H. Lodhi, C. Saunders, J. Shawe-Taylor, N. Cristianini, and C. Watkins, “Text classification using string kernels,” *J. Mach. Learn. Res.*, vol. 2, no. 30, pp. 419–444, Mar. 2002.

[105] N. Cancedda, E. Gaussier, C. Goutte, and J.-M. Renders, “Word-Sequence Kernels,” *J. Mach. Learn. Res.*, vol. 3, no. 6, pp. 1059–1082, Aug. 2003.

[106] A. Moschitti and S. Quarteroni, “Kernels on Linguistic Structures for Answer Extraction,” in *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics on Human Language Technologies: Short Papers*, 2008, no. June, pp. 113–116.

[107] Y. Chali, S. A. Hassan, and I. Kaisar, “Using Syntactic and Shallow Semantic Kernels to Improve Multi-Modality Manifold-Ranking for Topic-Focused Multi-Document Summarization,” in *Proceddings of 5th International Joint Conference on Natural Language Processing*, 2011, pp. 1098–1106.

[108] R. C. Bunescu and R. J. Mooney, “A Shortest Path Dependency Kernel for Relation Extraction,” in *Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing*, 2005, no. October, pp. 724–731.

[109] A. Culotta and J. Sorensen, “Dependency tree kernels for relation extraction,” in *Proceedings of the 42nd Annual Meeting on Association for Computational Linguistics*, 2004, no. Table 2.

[110] D. K. Evans, J. L. Klavans, and K. R. Mckeown, “Columbia Newsblaster : Multilingual News Summarization on the Web,” in *Proceedings of Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics,* 2004, pp. 1–4.

[111] X. Cai and W. Li, “A spectral analysis approach to document summarization: Clustering and ranking sentences simultaneously,” *Inf. Sci. (Ny).*, vol. 181, no. 18, pp. 3816–3827, Sep. 2011.

[112] D. Wang, S. Zhu, T. Li, Y. Chi, and Y. Gong, “Integrating Document Clustering and Multidocument Summarization,” *ACM Trans. Knowl. Discov. Data*, vol. 5, no. 3, pp. 1–26, Aug. 2011.

[113] K. Woodsend and M. Lapata, “Automatic Generation of Story Highlights,” in *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, 2010, no. July, pp. 565–574.

[114] K. M. Svore, M. Way, L. Vanderwende, and C. J. C. Burges, “Enhancing Single-document Summarization by Combining RankNet and Third-party Sources,” in *Proceedings of the 2007 joint conference on empirical methods in natural language processing and computational natural language learning (EMNLP-CoNLL 2007)*, 2007, pp. 448–457.

[115] W. T. Chuang and J. Yang, “Extracting Sentence Segments for Text Summarization : A Machine Learning Approach,” in *Proceedings of the 23rd annual international ACM SIGIR conference on Research and development in information retrieval*, 2000, pp. 152–159.

[116] Z. Cao, F. Wei, L. Dong, S. Li, and M. Zhou, “Ranking with recursive neural networks and its application to multi-document summarization,” in *Twenty-Ninth AAAI Conference on Artificial Intelligence*, 2015.

[117] K. Knight and D. Marcu, “Summarization beyond sentence extraction : A probabilistic approach to sentence compression,” *Artif. Intell.*, vol. 139, no. 1, pp. 91–107, 2002.

[118] M. Galley and K. R. Mckeown, “Lexicalized Markov Grammars for Sentence Compression,” in *Proceedings of the NAACL/HLT*, 2005, pp. 180–187.

[119] T. Cohn and M. Lapata, “Sentence Compression as Tree Transduction,” *J. Artif. Intell. Res.*, vol. 34, no. 1, pp. 637–674, 2009.

[120] J. Clarke and M. Lapata, “Global Inference for Sentence Compression : An Integer Linear Programming Approach Doctor of Philosophy School of Informatics University of Edinburgh,” *J. Artif. Intell. Res.*, vol. 31, pp. 399–429, 2008.

[121] J. Clarke and M. Lapata, “Discourse Constraints for Document Compression,” *Comput. Linguist.*, vol. 36, no. March, pp. 411–441, 2010.

[122] T. Berg-Kirkpatrick, D. Gillick, and D. Klein, “Jointly Learning to Extract and Compress,” in *Proc. of ACL*, 2011, pp. 481–490.

[123] C. Li, F. Liu, F. Weng, and Y. Liu, “Document Summarization via Guided Sentence Compression,” in *Empirical Methods in Natural Language Processing*, 2013, no. October, pp. 490–500.

[124] M. Almeida and A. Martins, “Fast and robust compressive summarization with dual decomposition and multi-task learning,” in *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics*, 2013, pp. 196–206.

[125] X. Qian and Y. Liu, “Fast Joint Compression and Summarization via Graph Cuts,” in *Empirical Methods in Natural Language Processing*, 2013, no. October, pp. 1492–1502.

[126] E. Marsi and E. Krahmer, “Explorations in sentence fusion,” in *Proceedings of the 10th European Workshop on Natural Language Generation*, 2005.

[127] K. Filippova and M. Strube, “Sentence Fusion via Dependency Graph Compression,” *Comput. Linguist.*, no. October, pp. 177–185, 2008.

[128] K. Mckeown, S. Rosenthal, K. Thadani, and C. Moore, *Time-Efficient Creation of an Accurate Sentence Fusion Corpus*, no. June. Association for Computational Linguistics, 2010.

[129] M. Elsner and D. Santhanam, “Learning to fuse disparate sentences,” in *Proceedings of the Workshop on Monolingual Text-To-Text Generation*, 2011, pp. 54–63.

[130] K. Thadani and K. Mckeown, “Supervised Sentence Fusion with Single-Stage Inference,” *Proc. IJCNLP 2013*, no. October, pp. 1410–1418, 2013.

[131] J. Chi, K. Cheung, and G. Penn, “Unsupervised Sentence Enhancement for Automatic Summarization,” *Emnlp*, pp. 775–786, 2014.

[132] E. Tzouridis, J. A. Nasir, and U. Brefeld, “Learning to Summarise Related Sentences,” in *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics*, 2014, pp. 1636–1647.

[133] A. Khan, N. Salim, and Y. Jaya Kumar, “A framework for multi-document abstractive summarization based on semantic role labelling,” *Appl. Soft Comput.*, vol. 30, pp. 737–747, May 2015.

[134] J. A. Motta, “Insertion of Ontological Knowledge to Improve Automatic Summarization Extraction Methods,” *J. Intell. Learn. Syst. Appl.*, vol. 03, no. 03, pp. 131–138, 2011.

[135] Y. Gong and X. Liu, “Generic Text Summarization Using Relevance Measure and Latent Semantic Analysis,” in *Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval*, 2002, pp. 19–25.

[136] B. Hachey, G. Murray, and D. Reitter, “The Embra System at DUC 2005 : Query-oriented Multi-document Summarization with a Very Large Latent Semantic Space,” in *Proceedings of the Document Understanding Conference*, 2005.

[137] J. Steinberger, M. Poesio, M. a. Kabadjov, and K. Ježek, “Two uses of anaphora resolution in summarization,” *Inf. Process. Manag.*, vol. 43, no. 6, pp. 1663–1680, Nov. 2007.

[138] E. D. Yirdaw and D. Ejigu, “Topic-based Amharic text summarization with probabilistic latent semantic analysis,” in *Proceedings of the International Conference on Management of Emergent Digital EcoSystems - MEDES ’12*, 2012, p. 8.

[139] J.-H. Lee, S. Park, C.-M. Ahn, and D. Kim, “Automatic generic document summarization based on non-negative matrix factorization,” *Inf. Process. Manag.*, vol. 45, no. 1, pp. 20–34, Jan. 2009.

[140] D. Wang, T. Li, S. Zhu, and C. Ding, “Multi-document summarization via sentence-level semantic analysis and symmetric matrix factorization,” in *Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval - SIGIR ’08*, 2008, p. 307.

[141] I. V. Mashechkin, M. I. Petrovskiy, D. S. Popov, and D. V. Tsarev, “Automatic text summarization using latent semantic analysis,” *Program. Comput. Softw.*, vol. 37, no. 6, pp. 299–305, 2011.

[142] D. Wang, C. Ding, and T. Li, “Feature subset non-negative matrix factorization and its applications to document understanding,” in *Proceeding of the 33rd international ACM SIGIR*

conference on Research and development in information retrieval - SIGIR '10, 2010, p. 805.

[143] J. Steinberger, M. Turchi, M. Kabadjov, and R. Steinberger, "Wrapping up a Summary : from Representation to Generation," in *ACL 2010 Conference Short Papers*, 2010, no. July, pp. 382–386.

[144] G. Tsatsaronis, I. Varlamis, and K. Nørvåg, "SemanticRank: ranking keywords and sentences using semantic graphs," in *COLING '10 Proceedings of the 23rd International Conference on Computational Linguistics*, 2010, no. August, pp. 1074–1082.

[145] Z. Shi, G. Melli, Y. Wang, Y. Liu, B. Gu, M. M. Kashani, A. Sarkar, and F. Popowich, "Question Answering Summarization of Multiple Biomedical Documents," *Proc. 20th Can. Conf. Aritificial Intell. (CanAI '07)*, pp. 284–295, 2007.

[146] L. P. Morales, A. D. Esteban, P. Gervás, I. Matveeva, C. Biemann, M. Choudhury, and M. Diab, "Concept-graph based biomedical automatic summarization using ontologies," in *Proceedings of the 3rd Textgraphs Workshop on Graph-Based Algorithms for Natural Language Processing*, 2008, no. August, pp. 53–56.

[147] Y. Sankarasubramaniam, K. Ramanathan, and S. Ghosh, "Text summarization using Wikipedia," *Inf. Process. Manag.*, vol. 50, no. 3, pp. 443–461, 2014.

[148] J. Leskovec, "Learning Sub-structures of Document Semantic Graphs for Document Summarization," in *KDD Workshop on Link Analysis and Group Detection*, 2004, no. August.

[149] J. Leskovec, N. Milic-Frayling, and M. Grobelnik, "Impact of linguistic analysis on the semantic graph coverage and learning of document extracts," in *Proceedings of the 20th national conference on Artificial intelligence*, 2005, pp. 1069–1074.

[150] J. Leskovec, N. Milic-Frayling, and M. Grobelnik, "Extracting Summary Sentences Based on the Document Semantic Graph," *Microsoft Tech. Rep. TR-2005-07*, 2005.

[151] L. Plaza, A. Díaz, and P. Gervás, "A semantic graph-based approach to biomedical summarisation.," *Artif. Intell. Med.*, vol. 53, no. 1, pp. 1–14, Sep. 2011.

[152] G. Glavaš and J. Šnajder, "Event graphs for information retrieval and multi-document summarization," *Expert Syst. Appl.*, vol. 41, no. 15, pp. 6904–6916, Nov. 2014.

[153] S. SUDHAHAR, G. DE FAZIO, R. FRANZOSI, and N. CRISTIANINI, "Network analysis of narrative content in large corpora.," *Nat. Lang. Eng.*, vol. 21, pp. 81–112, 2015.

[154] S. Yan and X. Wan, "Deep Dependency Substructure-Based Learning for Multidocument Summarization," *ACM Trans. Inf. Syst.*, vol. 34, no. 1, pp. 1–24, Jul. 2015.

[155] P.-E. Genest and G. Lapalme, "Fully abstractive approach to guided summarization," in *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Short Papers - Volume 2*, 2012, pp. 354–358.

[156] E. Lloret and M. Palomar, "Challenging Issues of Automatic Summarization : Relevance Detection and Quality-based Evaluation," *Inform.*, vol. 34, pp. 29–35, 2010.

[157] C. Lin and M. Rey, "R OUGE : A Package for Automatic Evaluation of Summaries," in *Proceedings of the ACL-04 Workshop: Text Summarization Branches Out*, 2004, pp. 74–81.

[158] a Nenkova and R. Passonneau, "Evaluating content selection in summarization: The pyramid method," *Proc. HLT-NAACL*, vol. 2004, pp. 145–152, 2004.

[159] R. J. Passonneau, A. Nenkova, K. McKeown, and S. Sigelman, "Applying the pyramid method in DUC 2005," in *Proceedings of the Document Understanding Conference (DUC 05), Vancouver, BC, Canada*, 2005.

[160] R. J. Passonneau, "Formal and Functional Assessment of the Pyramid Method for Summary Content Evaluation," *Nat. Lang. Eng.*, vol. 16, no. 02, pp. 107–131, 2010.

[161] J. Steinberger and K. Jezek, “Evaluation Measures for Text Summarization,” *Comput. Informatics*, vol. 28, pp. 1001–1025, 2009.

[162] S. Mackie, R. Mccreadie, C. Macdonald, and I. Ounis, “On Choosing an Effective Automatic Evaluation Metric for Microblog Summarisation,” in *Proceedings of the 5th Information Interaction in Context Symposium*, 2014, pp. 115–124.

[163] A. S. Eneko Agirre, Oier López de Lacalle, “Random walks for knowledge-based word sense disambiguation,” *Diss. Abstr. Int. B Sci. Eng.*, vol. 70, no. 8, p. 4943, 2014.

[164] K. Owczarzak, J. M. Conroy, H. T. Dang, and A. Nenkova, “An Assessment of the Accuracy of Automatic Evaluation in Summarization,” in *Proceedings of Workshop on Evaluation Metrics and System Comparison for Automatic Summarization*, 2012, pp. 1–9.

[165] R. Katragadda, “On alternative automated content evaluation measures,” *Evaluation*, pp. 1–10.

[166] A. Louis and a Nenkova, “Predicting summary quality using limited human input,” *Proc. TAC*, 2009.

[167] P. Over, H. Dang, and D. Harman, “DUC in context,” *Inf. Process. Manag.*, vol. 43, no. 6, pp. 1506–1520, Nov. 2007.

[168] H. Bouamor, A. Max, and A. Vilnat, “Multitechnique paraphrase alignment: A contribution to pinpointing sub-sentential paraphrases,” *ACM Trans. Intell. Syst. Technol. - Spec. Sect. Paraphrasing; Intell. Syst. Soc. Aware Comput. Soc. Comput. Behav. Model. Predict.*, vol. 4, no. 3, 2013.

[169] W. a Gale and W. Church, Kenneth, “A Program for Aligning Sentences in Bilingual Corpora,” *Comput. Linguist.*, vol. 19, pp. 177–184, 1993.

[170] D. Wu, “Aligning a parallel English-Chinese corpus statistically with lexical criteria,” in *Proceedings of the 32nd annual meeting on Association for Computational Linguistics -*, 1994, pp. 80–87.

[171] M. Haruno and T. Yamazaki, “High-performance bilingual text alignment using statistical and dictionary information,” *Nat. Lang. Eng.*, vol. 3, no. 1, pp. 1–14, 1997.

[172] M. Kay and M. Roscheisen, “Text-translation alignment,” *Comput. Linguist.*, vol. 19, no. 1, pp. 121–142, 1993.

[173] S. F. Chen, “Aligning Sentences in Bilingual Corpora Using Lexical Information,” in *Proceedings of the 31st annual meeting on Association for Computational Linguistics*, 93AD, vol. XXXIII, no. 2, pp. 9–16.

[174] A. Meyers, R. Yangarber, and R. Grishman, “Alignment of shared forests for bilingual corpora,” in *Proceedings of the 16th conference on Computational linguistics*, 1996, pp. 460–65.

[175] R. Barzilay and N. Elhadad, “Sentence alignment for monolingual comparable corpora,” in *Proceedings of the 2003 conference on Empirical methods in natural language processing*, 2003, pp. 25–32.

[176] S. Martens and V. Vincent, “An efficient, generic approach to extracting multi-word expressions from dependency trees,” in *CoLing Workshop: Multiword Expressions: From Theory to Applications (MWE 2010)*, 2010.

[177] M.-C. de Marneffe, T. Dozat, N. Silveira, K. Haverinen, F. Ginter, J. Nivre, and C. D. Manning, “Universal Stanford Dependencies: A cross-linguistic typology,” *Proc. Ninth Int. Conf. Lang. Resour. Eval.*, pp. 4585–4592, 2014.

[178] G. Puscasu, “A Multilingual Method for Clause Splitting,” in *Proceedings of the 7th Annual*

Colloquium for the UK Special Interest Group for Computational Linguistics, 2004.

[179] X. Carreras, L. Màrquez, and J. Castro, "Filtering-ranking perceptron learning for partial parsing," *Mach. Learn.*, vol. 60, no. 1–3, pp. 41–71, 2005.

[180] V. Van Nguyen, M. Le Nguyen, and A. Shimazu, "Using conditional random fields for clause splitting," in *Proceedings of The Pacific Association for Computational Linguistics*, 2007, pp. 58–65.

[181] A. Bies, M. Ferguson, K. Katz, R. MacInture, V. Tredinnick, G. Kim, M. A. Marcinkiewicz, and B. Schasberger, "Bracketing guidelines for Treebank II style Penn Treebank project," 1995.

[182] K. Ganesan, C. Zhai, and J. Han, "Opinosis : A Graph-Based Approach to Abstractive Summarization of Highly Redundant Opinions," in *Proceedings of the 23rd International Conference on Computational Linguistics*, 2010, no. August, pp. 340–348.

[183] I. F. Moawad and M. Aref, "Semantic graph reduction approach for abstractive Text Summarization," *2012 Seventh Int. Conf. Comput. Eng. Syst.*, pp. 132–138, Nov. 2012.

[184] A. dAcierno, V. Moscato, F. Persia, A. Picariello, and A. Penta, "iWIN: A Summarizer System Based on a Semantic Analysis of Web Documents," *2012 IEEE Sixth Int. Conf. Semant. Comput.*, pp. 162–169, 2012.

[185] J. Kummerfeld, D. Hall, J. Curran, and D. Klein, "Parser showdown at the wall street corral: an empirical investigation of error types in parser output," in *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, 2012, no. July, pp. 1048–1059.

[186] K. Toutanova, D. Klein, C. D. Manning, and Y. Singer, "Feature-rich part-of-speech tagging with a cyclic dependency network," in *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology - NAACL '03*, 2003, vol. 1, pp. 173–180.

[187] J. R. Finkel, T. Grenager, and C. Manning, "Incorporating Non-local Information into Information Extraction Systems by Gibbs Sampling," in *43nd Annual Meeting of the Association for Computational Linguistics*, 2005, no. June, pp. 363–370.

[188] S. Brin and L. Page, "The anatomy of a large-scale hypertextual Web search engine BT - Computer Networks and ISDN Systems," *Comput. Networks ISDN Syst.*, vol. 30, no. 1/7, pp. 107–117, 1998.

[189] M. Franceschet, "PageRank," *Commun. ACM*, vol. 54, no. 6, p. 92, 2011.

[190] E. Krahmer and K. van Deemter, "Computational Generation of Referring Expressions: A Survey," *Comput. Linguist.*, vol. 38, no. 1, pp. 173–218, 2012.

[191] A. Nenkova and K. McKeown, "References to named entities," in *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology companion volume of the Proceedings of HLT-NAACL 2003--short papers - NAACL '03*, 2003, vol. 2, pp. 70–72.

[192] C. Larman, *Applying UML and Patterns: An Introduction to Object-Oriented Analysis and Design and the Unified Process (2nd Edition)*. Prentice Hall PTR, 2001.

[193] R. J. Abbott, "Program design by informal English descriptions," *Communications of the ACM*, vol. 26, no. 11. pp. 882–894, 1983.

[194] W. Cyre, "A requirements sublanguage for automated analysis," *Int. J. Intell. Syst.*, vol. 10, no. 7, pp. 665–689, 1995.

[195] N. E. Fuchs and R. Schwitter, "Specifying Logic Programs in Controlled Natural Language," in *Workshop on Computational Logic for Natural Language Processing*, 1995.

[196] M. Elbendak, P. Vickers, and N. Rossiter, “Parsed use case descriptions as a basis for object-oriented class model generation,” *J. Syst. Softw.*, vol. 84, no. 7, pp. 1209–1223, 2011.

[197] V. B. R. Vidya Sagar and S. Abirami, “Conceptual modeling of natural language functional requirements,” *J. Syst. Softw.*, vol. 88, no. 1, pp. 25–41, 2014.

[198] J. F. Sowa, “Conceptual graphs as a universal knowledge representation,” *Computers & Mathematics with Applications*, vol. 23, no. 2–5. pp. 75–93, 1992.

[199] J. F. Sowa, *Handbook of Knowledge Representation*, vol. 3. Elsevier, 2008.

[200] S. Y. Yang and V. W. Soo, “Extract conceptual graphs from plain texts in patent claims,” *Eng. Appl. Artif. Intell.*, vol. 25, no. 4, pp. 874–887, 2012.

[201] P. R. K. Rao and S. L. Devi, “Automatic Identification of Conceptual Structures using Deep Boltzmann Machines,” in *Proceedings of the 7th Forum for Information Retrieval Evaluation*, 2015, pp. 0–4.

[202] S. Hensman, “Construction of conceptual graph representation of texts,” pp. 49–54, May 2004.

[203] B. Kamsu-Foguem, G. Tchuenté-Foguem, and C. Foguem, “Conceptual graph operations for formal visual reasoning in the medical domain,” *IRBM*, 2014.

[204] B. Motik, P. F. Patel-Schneider, and B. Parsia, “OWL 2 Web Ontology Language,” in *OWL 2 Web Ontology Language Primer*, 2009, pp. 196–205.

[205] D. Gerber, S. Hellmann, Lorenz Bühmann, T. Soru, R. Usbeck, and A.-C. N. Ngomo, “Real-time RDF extraction from unstructured data streams.,” in *The Semantic Web–ISWC 2013*, 2013, pp. 135–150.

[206] L. J. Garcia Castro, C. McLaughlin, and A. Garcia, “Biotea: RDFizing PubMed Central in support for the paper as an interface to the Web of Data.,” *J. Biomed. Semantics*, vol. 4 Suppl 1, no. Suppl 1, p. S5, 2013.

[207] C. Boitet, “A rationale for using UNL as an interlingua and more in various domains,” *Res. Comput. Sci.*, vol. 12, pp. 3–9, 2005.

[208] I. Boguslavsky, N. Frid, L. Iomdin, L. Kreidlin, I. Sagalova, and V. Sizov, “Creating a Universal Networking Language module within an advanced NLP system,” in *Proceedings of the 18th conference on Computational linguistics -*, 2000, vol. 1, pp. 83–89.

[209] M. Sydow, M. Pikuła, and R. Schenkel, “The notion of diversity in graphical entity summarisation on semantic knowledge graphs,” *J. Intell. Inf. Syst.*, vol. 41, no. 2, pp. 109–149, 2013.

[210] N. Burton-Roberts, *Analysing Sentences*. Longman, 1997.

[211] C. D. Manning, M. Surdeanu, J. Bauer, J. Finkel, S. J. Bethard, and M. David, “The Stanford CoreNLP Natural Language Processing Toolkit,” in *Proceedings of 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, 2014, pp. 55–60.

[212] K. Filippova and Y. Altun, “Overcoming the Lack of Parallel Data in Sentence Compression,” in *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, 2013, pp. 1481–1491.

[213] J. Cordeiro, G. Dias, and P. Brazdil, “Unsupervised induction of sentence compression rules,” in *Proceedings of the 2009 Workshop on …*, 2009, no. August, pp. 15–22.

[214] J. Turner, J. Turner, E. Charniak, and E. Charniak, “Supervised and Unsupervised Learning for Sentence Compression,” in *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL’05)*, 2005, no. June, pp. 290–297.

[215] T. Berg-Kirkpatrick, D. Gillick, and D. Klein, “Jointly Learning to Extract and Compress,” in *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies - Volume 1}, series HLT ’11*, 2011, pp. 481–490.

[216] P. Genest and G. Lapalme, “Framework for Abstractive Summarization using Text-to-Text Generation,” no. June, pp. 64–73, 2011.

[217] A. F. T. Martins and N. a. Smith, “Summarization with a joint model for sentence extraction and compression,” in *NAACL HLT Workshop on Integer Linear Programming for Natural Langauge Processing - ILP ’09*, 2009, no. June, pp. 1–9.

[218] S. Brin and L. Page, “The anatomy of a large-scale hypertextual Web search engine,” *Comput. Networks ISDN Syst.*, vol. 30, no. 1–7, pp. 107–117, Apr. 1998.

[219] H. Van Halteren, “University of Nijmegen.Writing style recognition and sentence extraction,” in *DUC 2002*, 2002, pp. 66–70.

[220] M. Brunn and B. Dufour, “The University of Lethbridge Text Summarizer at DUC 2002,” in *DUC 2002*, 2002, no. c, pp. 1–6.

[221] R. Angheluta, R. De Busser, and M. Moens, “The Use of Topic Segmentation for Automatic Summarization,” in *DUC 2002*, 2002.

[222] T. Copeck, S. Szpakowicz, and N. Japkowic, “Learning How Best to Summarize,” in *Workshop on Text Summa- rization (In conjunction with the ACL 2002 and including the DARPA/NIST sponsored DUC 2002 Meeting on Text Summarization)*, 2002.

[223] M. Karamuftuoglu, “An approach to summarisation based on lexical bonds,” in *DUC2002*, 2002, vol. 2, pp. 86–89.

[224] J. Otterbach, A. Winkel, and D. Radev, “The Michigan Single and Multi-document Summarizer for DUC 2002,” in *Duc 2002*, 2002.

[225] P. Lal and S. Riiger, “Extract-based Summarization with Simplification,” in *DUC 2002*, 2002.

[226] H. Daumé, A. Echihabi, D. Marcu, D. Munteanu, and R. Soricut, “GLEANS: A generator of logical extracts and abstracts for nice summaries,” in *DUC 2002*, 2002.

[227] A. Farzindar, H. Saggion, and G. Lapalme, “Summaries with SumUM and its Expansion for Document Understanding Conference,” in *DUC 2002*, 2002.

[228] W. Kraaij, M. Spitters, and A. Hulth, “Headline extraction based on a combination of uni- and multidocument summarization techniques,” in *DUC 2002*, 2002.

[229] M. Joshi, H. Wang, and S. Mcclean, “Dense Semantic Graph and its Application in Single Document Summarisation Previous Work on Graph based Text Summarisation,” in *DART@AI*IA 2013*, pp. 25–36.

[230] V. Rus, M. Lintean, R. Banjade, N. Niraula, and D. Stefanescu, “SEMILAR : The Semantic Similarity Toolkit,” in *ACL 2013*, 2013, pp. 163–168.

[231] J. Lin, “Divergence Measures on the Shannon Entropy,” *IEEE Trans. Inf. Theory*, vol. 37, no. I, pp. 145–151, 1991.

[232] L. Shi, M. Zhou, "Improved sentence alignment on parallel web pages using a stochastic tree alignment model", in Proceedings of the Conference on Empirical Methods in Natural Language Processing,2008, pp. 505–513. *ACL 2008*, Honolulu